

K 13

FILE ID**INITAP

```
1 0001 0 MODULE INITAP (
2 0002 0   LANGUAGE (BLISS32),
3 0003 0   IDENT = 'V04-000'
4 0004 0   )
5 0005 1 BEGIN
6 0006 1
7 0007 1
8 0008 1 ****
9 0009 1 *
10 0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
11 0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
12 0012 1 * ALL RIGHTS RESERVED.
13 0013 1 *
14 0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
15 0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
16 0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
17 0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
18 0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
19 0019 1 * TRANSFERRED.
20 0020 1 *
21 0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
22 0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
23 0023 1 * CORPORATION.
24 0024 1 *
25 0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
26 0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
27 0027 1 *
28 0028 1 *
29 0029 1 ****
30 0030 1 *
31 0031 1 ++
32 0032 1
33 0033 1 FACILITY: INIT Utility Structure Level II
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1
37 0037 1     THIS MODULE HANDLES INITIALIZATION OF ANSI MAGNETIC TAPE
38 0038 1
39 0039 1 ENVIRONMENT:
40 0040 1
41 0041 1     VAX/VMS operating system, including privileged system services
42 0042 1     and internal exec routines.
43 0043 1
44 0044 1 --
45 0045 1
46 0046 1
47 0047 1     AUTHOR: D. H. GILLESPIE.      CREATION DATE: 10-DEC-1977 18:10
48 0048 1
49 0049 1 MODIFIED BY:
50 0050 1
51 0051 1     V03-011 MMD0269      Meg Dumont, 23-Mar-1984 9:19
52 0052 1     Change the processing of the accessibility character fields
53 0053 1     in the VOL1 and/or HDR1 label to call the installation
54 0054 1     specific accessibility routine. The return from this
55 0055 1     routine determines the user's access to the volume and/or file.
56 0056 1
57 0057 1     V03-009 MMD0237      Meg Dumont, 14-Feb-1984 11:16
```

58 0058 1 | Change all calls to CLEAR_VALID to a QIO IOS_AVAILABLE.
59 0059 1 | Delete any reference to SET_VALID.
60 0060 1 |
61 0061 1 | V03-008 MCN0140 Maria del C. Nasr 30-Nov-1983
62 0062 1 | Define LABEL STRING as BBLOCK descriptor and use
63 0063 1 | descriptor offsets to find length.
64 0064 1 |
65 0065 1 | V03-007 MMD0180 Meg Dumont, 26-May-1983 15:15
66 0066 1 | Change VOL1 to indicate ANSI level 4 when writing a SYSTEM
67 0067 1 | CODE in the VOL1 label.
68 0068 1 |
69 0069 1 | V03-006 STJ3091 Steven T. Jeffreys, 27-Apr-1983
70 0070 1 | Added support for /[NO]ERASE.
71 0071 1 |
72 0072 1 | V03-005 MMD0133 Meg Dumont, 12-Apr-1983 17:20
73 0073 1 | Turn on support for writing the VOL1 OWNER IDENTIFIER
74 0074 1 | field so that it is now a nonVMS field. Add support
75 0075 1 | for the underscore as a valid character to tape.
76 0076 1 |
77 0077 1 | V03-004 MMD0127 Meg Dumont, 1-Apr-1983 14:13
78 0078 1 | Fix to the temp fix
79 0079 1 |
80 0080 1 | V03-003 MMD0126 Meg Dumont, 1-Apr-1983 13:28
81 0081 1 | Temp take out references to VOL_OWNER
82 0082 1 |
83 0083 1 | V03-002 MMD0117 Meg Dumont, 29-Mar-1983 0:42
84 0084 1 | Add support for new VMS protection. Means writing a
85 0085 1 | VOL2 label to the tape when a VMS protection is specified
86 0086 1 |
87 0087 1 | V03-001 MMD0001 Meg Dumont, 13-Aug-1982 13:11
88 0088 1 | Change from call to SET_VALID to QIO IOS_PACKACK
89 0089 1 |
90 0090 1 | V02-013 DMW0018 David Michael Walp 2-Mar-1982
91 0091 1 | Another correction for the volume invalid problem
92 0092 1 |
93 0093 1 | V02-012 DMW0016 David Michael Walp 18-Dec-1981
94 0094 1 | Increase Transtable size to 256
95 0095 1 |
96 0096 1 | V02-011 DMW0011 David Michael Walp 21-Aug-1981
97 0097 1 | Correct override typo and new Tape_Own_Prot and
98 0098 1 | /LABEL for /ANSI
99 0099 1 |
100 0100 1 | V02-010 DMW0010 David Michael Walp 18-Jun-1981
101 0101 1 | Cleaned up defaulting of density.
102 0102 1 |
103 0103 1 | V02-009 DMW0009 David Michael Walp 19-May-1981
104 0104 1 | Placed Volume Id into the File Set Id of the 'Dummy File'
105 0105 1 |
106 0106 1 | V02-008 DMW0008 David Michael Walp 1-May-1981
107 0107 1 | Upcased Volume label and check for illegal (non ANSI 'a'
108 0108 1 | characters)
109 0109 1 |
110 0110 1 | V02-007 DMW0006 David Michael Walp 25-Apr-1981
111 0111 1 | Created routine SET_CHARACTER (reset parity and
112 0112 1 | format)
113 0113 1 |
114 0114 1 | V02-006 DMW0004 David Michael Walp 9-Apr-1981

115 0115 1 | Added switch '/ANSI=VOLUME_ACCESSIBILITY:"x"'
116 0116 1 | Fixed bugs with override switches and error returns
117 0117 1 | Created FORMAT_VOL1 from old and new code
118 0118 1 | Reformatted module
119 0119 1 |
120 0120 1 | V02-005 DMW0001 David Michael Walp 10-Dec-1980
121 0121 1 | Replace Check_Prot procedure. Old procedure was
122 0122 1 | confused by the fact that init was installed with sysprv
123 0123 1 | for version 2.
124 0124 1 |
125 0125 1 | V02-004 RLRDENS Robert L. Rappaport 8-Oct-1980
126 0126 1 | At the same time that /DENSITY=1 and /DENSITY=2 support
127 0127 1 | is being added to INITIALIZE, we correct the problem
128 0128 1 | of INITIALIZE returning SSS_VOLINV when the INITIALIZE
129 0129 1 | follows a DISMOUNT/NOUNLOAD in a command procedure.
130 0130 1 |
131 0131 1 | V02-003 MCN0001 Maria del C. Nasr, 20-Jun-1980 15:10
132 0132 1 | Change DECFILE112 to DECFILE11A in HDR1, and eliminate binary
133 0133 1 | data from HDR2. This is part of the implementation of HDR3.
134 0134 1 |
135 0135 1 | V0100 ACG00001 Andrew C. Goldstein, 10-Oct-1978 21:27
136 0136 1 | Previous revision history moved to [INIT.SRC]INIT.REV
137 0137 1 | **
138 0138 1 |
139 0139 1 |
140 0140 1 LIBRARY 'SYSSLIBRARY:LIB:L32';
141 0141 1 REQUIRE 'SRCS:INIDEF.B32';
142 0432 1 REQUIRE 'LIBDS:[VMSLIB.OBJ]INITMSG.B32';
143 0564 1 FORWARD ROUTINE
144 0565 1 CHECK PROT,
145 0566 1 DEFAULT_CHAR : NOVALUE, | check volume protection
146 0567 1 | set default characteristic
147 0568 1 | of tape drive
148 0569 1 FORMAT_VOL1_VOL2,
149 0570 1 INIT_TAPE : NOVALUE, | format the VOL1 and VOL2 label
150 0571 1 READ_VOLLABELS : NOVALUE, | main control for tape init
151 0572 1 | read & verify VOL1 & HDR1
152 0573 1 SET_DENSITY : NOVALUE: | ANSI labels
153 0574 1 | set the density of the drive
154 0575 1 EXTERNAL ROUTINE
155 0576 1 CALDAYNO, | calculate day number (chop
156 0577 1 | hour min sec from binary)
157 0578 1 CONVDATE_J2R, | convert date ANSI tape JULIAN
158 0579 1 | to VMS
159 0580 1 CONVDATE_R2J, | convert VMS date to ANSI
160 0581 1 | JULIAN format on tape
161 0582 1 GET_CHANNELUCB, | Given channel number get assoc UCB
162 0583 1 GET_RECORD, | get current record drive is reading
163 0584 1 WRITE_USER_UVL, | write user volume labels
164 0585 1 FORMAT_VOLOWNER : NOVALUE, | format volume owner field
165 0586 1 LIBSCVT_OTB : ADDRESSING_MODE(ABSOLUTE),
166 0587 1 PROCESS_VOL2_LABEL,
167 0588 1 TAPE_OWN_PROT: | process the VOL2 label
168 0589 1 | determine protection and
169 0590 1 | owner of tape
170 0591 1 EXTERNAL CHANNEL, | channel of volume
171 0592 1 |

: 172 0593 1 CTLSGQ PROCPRIV : REF BBLOCK ADDRESSING_MODE(ABSOLUTE),
: 173 0594 1 INIT OPTIONS : BITVECTOR, ! init option bits
: 174 0595 1 LABEL_STRING : BBLOCK [D\$C\$C_S_BLN], ! label descriptor
: 175 0596 1 OWNER_UIC, ! value of owner switch
: 176 0597 1 PROCESS_UIC, ! process uic
: 177 0598 1 PROTECTION, ! value of protection switch
: 178 0599 1 VOL_ACC : BYTE, ! value of label:volume switch
: 179 0600 1 VOL_OWNER : VECTOR [14,BYTE]; ! value of owner id field
: 180 0601 1
: 181 0602 1 BIND
: 182 0603 1 STARID = UPLIT ('DECFILE11A'); ! Set the value for VOL1 syscode
: 183 0604 1 OWN
: 184 0605 1 ANSI_LABEL : BBLOCK [80], ! ANSI label
: 185 0606 1 IO_STATUS : VECTOR [4,WORD], ! I/O status
: 186 0607 1 PRIVILEGE_MASK : REF BBLOCK, ! process privilege mask
: 187 0608 1 VOLUME PROT. ! protection for tape
: 188 0609 1 VOLUME_UIC, ! owner of tape
: 189 0610 1 ACCESS, ! users's access to magnetic tape
: 190 0611 1 CURRENT_RECORD, ! Tape record before call to \$MTACCESS
: 191 0612 1 LABEL_VER, ! ANSI label version decimal value
: 192 0613 1 UCB : REF BBLOCK, ! UCB address
: 193 0614 1 CHAR : VECTOR [4,BYTE], ! Char to output for tape accessibility
: 194 0615 1 VOL1 : BLOCK [80,BYTE]
: 195 0616 1 INITIAL(BYTE ('VOL1', ! VOL1 skeleton
: 196 0617 1 REP 75 OF BYTE(' ',
: 197 0618 1 '3')),
: 198 0619 1
: 199 0620 1 VOL2 : BLOCK [80,BYTE]
: 200 0621 1 INITIAL(BYTE ('VOL2', ! VOL2 skeleton
: 201 0622 1 'D\$C',
: 202 0623 1 REP 75 OF BYTE (' ')),
: 203 0624 1
: 204 0625 1 HDR1 : BLOCK [80,BYTE]
: 205 0626 1 INITIAL (BYTE ('HDR1', ! HDR1 skeleton
: 206 0627 1 REP 23 OF BYTE (' '),
: 207 0628 1 REP 3 OF BYTE ('0'),
: 208 0629 1 '1',
: 209 0630 1 REP 7 OF BYTE ('0'),
: 210 0631 1 '100',
: 211 0632 1 REP 15 OF BYTE (' '),
: 212 0633 1 REP 6 OF BYTE ('0'),
: 213 0634 1 'DECFILE11A',
: 214 0635 1 REP 8 OF BYTE (' ')),
: 215 0636 1
: 216 0637 1
: 217 0638 1 HDR2 : BLOCK[80,BYTE]
: 218 0639 1 INITIAL (BYTE('HDR2', ! HDR2 skeleton
: 219 0640 1 'F',
: 220 0641 1 REP 10 OF BYTE('0'),
: 221 0642 1 REP 35 OF BYTE(' '),
: 222 0643 1 '00',
: 223 0644 1 REP 28 OF BYTE(' '));
: 224 0645 1

```
: 226      0646 1 GLOBAL ROUTINE INIT_TAPE : NOVALUE =
: 227      0647 1
: 228      0648 1 | ++
: 229      0649 1
: 230      0650 1 | FUNCTIONAL DESCRIPTION:
: 231      0651 1 | This routine is the main control for tape initialization. If the
: 232      0652 1 | current tape is a valid files_11 ANSI tape, then the user must have
: 233      0653 1 | write privileges or be the owner of the tape. If the first file has
: 234      0654 1 | not expired, then the user must specify override expiration date and
: 235      0655 1 | have the privilege to do so. On new tapes the user must specify
: 236      0656 1 | to override both the expiration date and accessibility char in VOL1
: 237      0657 1 | and HDR1 and have VOLPRO priv to avoid the run away tape condition.
: 238      0658 1
: 239      0659 1 | CALLING SEQUENCE:
: 240      0660 1 |   INIT_TAPE()
: 241      0661 1
: 242      0662 1 | INPUT PARAMETERS:
: 243      0663 1 |   none
: 244      0664 1
: 245      0665 1 | IMPLICIT INPUTS:
: 246      0666 1 |   CLI parser database
: 247      0667 1
: 248      0668 1 | OUTPUT PARAMETERS:
: 249      0669 1 |   none
: 250      0670 1
: 251      0671 1 | IMPLICIT OUTPUTS:
: 252      0672 1 |   FILES-11 structure level II ansi magnetic tape initialized
: 253      0673 1
: 254      0674 1 | ROUTINE VALUE:
: 255      0675 1 |   none
: 256      0676 1
: 257      0677 1 | SIDE EFFECTS:
: 258      0678 1 |   none
: 259      0679 1
: 260      0680 1 | USER ERRORS:
: 261      0681 1 |   none
: 262      0682 1
: 263      0683 1 | --
: 264      0684 1
: 265      0685 2 | BEGIN
: 266      0686 2
: 267      0687 2 | LOCAL
: 268      0688 2 |   DESCRIPTOR : VECTOR [2],           | descriptor
: 269      0689 2 |   STATUS,          : VECTOR [12,BYTE],    | system service status
: 270      0690 2 |   TODAY,           :           ,           | buffer for today's date
: 271      0691 2 |   VMS_PROT;       :           ,           | VMS protection was specified
: 272      0692 2
: 273      0693 2 | EXTERNAL ROUTINE
: 274      0694 2 |   ERASE_BLOCKS;           ! erase the tape
: 275      0695 2
: 276      0696 2 | BIND
: 277      0697 2 |   SECONDS = UPLIT (-10000000,-1);   ! 1 second in 100 nsec units
: 278      0698 2
: 279      0699 2
: 280      0700 2 | ! The following note is left for historical reasons only!
: 281      0701 2 | ****
: 282      0702 2 | ! Here we have inserted a single QIO (IOS_REWIND) which apparently is not
```

283 0703 2 | needed but which in fact is here to take care of an anomaly that
284 0704 2 | sometimes occurs when the INITIALIZE command appears in a command file
285 0705 2 | immediately following a DISMOUNT/NOUNLOAD command.
286 0706 2 |
287 0707 2 | Under certain circumstances the INITIALIZE fails with a SSS VOLINV status.
288 0708 2 | The problem is due to a complicated interaction involving QIO dispatching
289 0709 2 | logic, the MAGTAPE ACP, and the INITIALIZE command. What occurs is the
290 0710 2 | following.
291 0711 2 |
292 0712 2 | DISMOUNT, before finishing issues a \$QIOW with an I/O function code of
293 0713 2 | IOS_ACPCONTROL!IOSM_DMOVE. This request is forwarded to the ACP and
294 0714 2 | DISMOUNT then has its image rundown.
295 0715 2 |
296 0716 2 | The ACP then issues a \$QIOW with a function code of IOS_REWIND!IOSM_NOWAIT,
297 0717 2 | while in parallel, INITIALIZE is starting up and it proceeds to set the
298 0718 2 | UCB\$M_VALID bit in UCB\$W_STS (which in this case was still on due to the
299 0719 2 | volume previously having been mounted) and then INITIALIZE issues its own
300 0720 2 | \$QIOW with an IOS_REWIND function code.
301 0721 2 |
302 0722 2 | In some instances, the ACP's REWIND QIO does not get as far as REQCOM
303 0723 2 | until after INITIALIZE's REWIND has been queued. If this occurs, INIT's
304 0724 2 | queued REWIND is started up before the ACP actually regains control and
305 0725 2 | the driver has no trouble since it finds the UCB\$M_VALID bit still on.
306 0726 2 | Unfortunately, as since as the ACP regains control, following the
307 0727 2 | driver's WFIKPCH, the ACP clears the UCB\$M_VALID bit. The next QIO
308 0728 2 | issued by INITIALIZE will fail due to the absence of the UCB\$M_VALID
309 0729 2 | bit.
310 0730 2 |
311 0731 2 | The solution (pronounced KLUDGE) herein implemented, simply inserts an extra
312 0732 2 | single \$QIOW with IOS_REWIND function code, surrounded by explicit
313 0733 2 | settings of the UCB\$M_VALID bit, before the real logic of INITIALIZE begins.
314 0734 2 | This \$QIOW allows the above potential interaction to occur, and after it is
315 0735 2 | finished, we again set the UCB\$M_VALID bit on.
316 0736 2 |*****
317 0737 2 |
318 0738 2 | The above is no longer true; that is we have eliminated the race condition
319 0739 2 | mentioned above by not doing issuing the rewind at dismount time
320 0740 2 | but instead marking the drive available. The following IO's mark
321 0741 2 | the volume valid then issue the rewind, which is necessary because
322 0742 2 | of the preMSCP drivers will not rewind on this function. The MSCP drivers
323 0743 2 | will and the second IO here becomes an NOP.
324 0744 2 |
325 0745 2 |
326 P 0746 2 STATUS = \$QIOW(
327 P 0747 2 CHAN = .CHANNEL,
328 P 0748 2 FUNC = IOS_PACKACK,
329 P 0749 2 IOSB = IO_STATUS[0]);
330 0750 2
331 P 0751 2 STATUS = \$QIOW(
332 P 0752 2 CHAN = .CHANNEL,
333 P 0753 2 FUNC = IOS_REWIND,
334 P 0754 2 IOSB = IO_STATUS[0]);
335 0755 2
336 0756 2 ! wait 10 seconds before giving up
337 0757 2
338 0758 2 INCR J FROM 0 TO 9 DO
339 0759 3 BEGIN

```
340 P 0760 3 STATUS = $QIOW(          :  
341 P 0761 3     CHAN = .CHANNEL,  
342 P 0762 3     FUNC = IOS_PACKACK,  
343 P 0763 3     IOSB = IO_STATUS[0];  
344 P 0764 3 STATUS = $QIOW(          :  
345 P 0765 3     CHAN = .CHANNEL,  
346 P 0766 3     FUNC = IOS_REWIND,  
347 P 0767 3     IOSB = IO_STATUS;  
348 P 0768 3 IF .STATUS THEN STATUS = .IO_STATUS[0];  
349 P 0769 4 IF .STATUS NEQ SSS_MEDOFL AND .STATUS NEQ SSS_VOLINV THEN EXITLOOP;  
350 P 0770 3 IF $SETIMR( DAYTIM=SECONDS, EFN = 0)  
351 P 0771 2 THEN SWAITFR( EFN = 0);  
352 P 0772 2 END;  
353 P 0773 2 ! all rewind errors reported to user  
354 P 0774 2  
355 P 0775 2 IF NOT .STATUS THEN ERR_EXIT(.STATUS);  
356 P 0776 2  
357 P 0777 2 ! set the VMS default tape drive characteristics  
358 P 0778 2  
359 P 0779 2 DEFAULT_CHAR();  
360 P 0780 2  
361 P 0781 2 ! check user access to rewrite ( DESTROY ) the tape  
362 P 0782 2  
363 P 0783 2 PRIVILEGE_MASK = CTL$GQ_PROCPRIV;           ! process privilege mask  
364 P 0784 2  
365 P 0785 2 ! Get the UCB associated with this channel  
366 P 0786 2  
367 P 0787 2 UCB = KERNEL_CALL(GET_CHANNELUCB,.CHANNEL);  
368 P 0788 2  
369 P 0789 2 ! The following check is here so that the operators has the ability  
370 P 0790 2 to bypass the first read to magnetic tape. This should be  
371 P 0791 2 used only when the magnetic tape is a blank tape. Blank tapes  
372 P 0792 2 are prone to run away conditions especially on some of the older  
373 P 0793 2 tape drives.  
374 P 0794 2  
375 P 0795 3 IF NOT (.INIT OPTIONS[OPT_OVR_EXP]          :  
376 P 0796 3 AND .INIT_OPTIONS[OPT_OVR_ACC]          : bypass all protection if  
377 P 0797 3 AND .INIT_OPTIONS[OPT_OVR_VOL0]          : override expiration and access  
378 P 0798 3 AND .PRIVILEGE_MASK[PRVSV_VOLPRO]          : characters, volume owner,  
379 P 0799 3 AND .PRIVILEGE_MASK[PRVSV_OPER])          : and volpro  
380 P 0800 2 THEN          : and oper  
381 P 0801 3 BEGIN  
382 P 0802 3     READ_VOLLABELS();                      ! is it an ANSI tape  
383 P 0803 3  
384 P 0804 3 ! If ACCESS is clear then we must give the user access to the tape  
385 P 0805 3 ! regardless of what the VMS protection specifies.  
386 P 0806 3  
387 P 0807 3 IF .ACCESS  
388 P 0808 3     THEN  
389 P 0809 4     BEGIN  
390 P 0810 5         IF (          :  
391 P 0811 6             (.INIT OPTIONS[OPT_OVR_EXP]          : does user have privilege  
392 P 0812 6             OR .INIT_OPTIONS[OPT_OVR_VOL0])          : or volume owner  
393 P 0813 6             AND NOT T.PRIVILEGE_MASK[PRVSV_VOLPRO] ! ( volpro priv or  
394 P 0814 6             OR .VOLUME_UIC EQL :PROCESS_UIC)          : owner of the tape )  
395 P 0815 5         )  
396 P 0816 4         OR
```

397 0817 5
398 0818 6
399 0819 5
400 0820 4
401 0821 3
402 0822 2
403 0823 2
404 0824 2
405 0825 2
406 0826 2
407 0827 2
408 0828 2
409 0829 2
410 0830 2
411 0831 2
412 0832 2
413 0833 2
414 0834 2
415 0835 2
416 0836 2
417 0837 2
418 0838 2
419 0839 2
420 0840 2
421 0841 2
422 0842 2
423 0843 2
424 0844 2
425 0845 2
426 0846 2
427 0847 2
428 0848 2
P 0849 2
P 0850 2
P 0851 2
P 0852 2
P 0853 2
P 0854 2
0855 2
P 0856 2
P 0857 2
P 0858 2
P 0859 2
P 0860 2
P 0861 2
P 0862 2
P 0863 2
P 0864 2
P 0865 2
P 0866 2
P 0867 2
P 0868 2
P 0869 2
P 0870 2
P 0871 2
P 0872 2
P 0873 2

```
(  
    NOT KERNEL_CALL (CHECK_PROT, .VOLUME_PROT, .VOLUME_UIC)  
    )  
    THEN ERR_EXIT(SSS_NOPRIV);  
END;  
END;  
  
! set default version number to 3, and format the volume label. Please  
note that if we write a VMS protectionon this tape then the LABEL_VER  
is set to 4, inside FORMAT_VOL1_VOL2.  
  
LABEL_VER = 3;  
VMS_PROT = (FORMAT_VOL1_VOL2());  
  
! default expiration and creation dates to today's date for HDR1  
DESCR[0] = 11;  
DESCR[1] = TODAY;  
SASCTIM(TIMBUF = DESCR);  
CONVDATE_R2J(TODAY,HDR1[HD1$T_CREATEDT]);  
CHSMOVE(AD1$S_CREATEDT,HDR1[HD1$T_CREATEDT],HDR1[HD1$T_EXPIREDT]);  
  
! Call the accessibility system service to get the character to output.  
First keep the record that the UCB is reading. The accessibility  
routine can not move the tape from under us! Thus we will compare  
this to the field after the call and if the tape was moved we punt  
the operation.  
  
CURRENT_RECORD = KERNEL_CALL(GET_RECORD,.UCB);  
  
CHAR = SMTACCESS(LBLNAM = 0,  
                  UIC = .PROCESS_UIC,  
                  STD VERSION = .LABEL_VER,  
                  ACCESS_CHAR = 0,  
                  ACCESS_SPEC = MTASK_NOCHAR,  
                  TYPE = MTASK_OUTHDT);  
  
STATUS = KERNEL_CALL(GET_RECORD,.UCB);  
IF .CURRENT_RECORD NEQ .STATUS  
    THEN ERR_EXIT(SSS_TAPEPOSLOST);  
  
HDR1[HD1$B_FILACCESS] = .CHAR[0];  
  
! write the file set id from the volume label, the MOUNT will place it  
in the MVL and the MTAACP will use it as the FILE SET ID  
move must be done after VOL1 has been set up, because Legal ANSI 'a'  
character check is in FORMAT_VOL1_VOL2  
  
CHSMOVE ( VL1$S_VOLLBL, VOL1[VL1$T_VOLLBL], HDR1[HD1$T_FILESETID] );  
  
! rewind the tape  
STATUS = $QIOW(  
                CHAN = .CHANNEL,  
                FUNC = IOS_REWIND,
```

```
454 0874 2 IOSB = IO_STATUS[0];  
455 0875 2  
456 0876 2 IF .STATUS THEN STATUS = .IO_STATUS[0]; ! report problems to user  
457 0877 2 IF NOT .STATUS THEN ERR_EXITT.STATUS);  
458 0878 2  
459 0879 2  
460 0880 2 ! set tape density if users has used /DENSITY qualifier  
461 0881 2  
462 0882 2 IF .INIT_OPTIONS [OPT_DENSITY] THEN SET_DENSITY ();  
463 0883 2  
464 0884 2  
465 0885 2 ! If the user requested it, erase the tape. This function is only valid  
466 0886 2 for the TU78 and MSCP tapes drives. All others will return SSS_ILLIOFUNC  
467 0887 2 to indicate that the hardware feature is not supported. Notify the user  
468 0888 2 if the erase did not happen. The operation of the erase is for the controller  
469 0889 2 to scribble on the tape starting from the current position and continuing to  
470 0890 2 the EOT mark, then rewinding to the BOT mark.  
471 0891 2  
472 0892 2 IF .INIT_OPTIONS [OPT_ERASE]  
473 0893 2 THEN  
474 0894 3 BEGIN  
475 0895 4 IF (STATUS = EXEC_CALL (ERASE_BLOCKS, 0, 1, .CHANNEL))  
476 0896 3 THEN  
477 0897 4 STATUS = .IO_STATUS[0];  
478 0898 3 IF NOT .STATUS  
479 0899 3 THEN  
480 0900 3 ERR_MESSAGE (INITS_ERASEFAIL, 0, .STATUS);  
481 0901 2 END;  
482 0902 2  
483 0903 2  
484 0904 2 ! now write VOL1 (UVL) HDR1 HDR2 ** EOF1 EOF2 ** in other words the volume  
485 0905 2 ! label and a dummy empty file ( so the label set are complete )  
486 0906 2  
487 P 0907 2 STATUS = $QIOW(  
488 P 0908 2     CHAN = .CHANNEL,  
489 P 0909 2     IOSB = IO_STATUS[0],  
490 P 0910 2     FUNC = IOS_WRITEBLK,  
491 P 0911 2     P1 = VOL1,  
492 P 0912 2     P2 = 80);  
493 P 0913 2 IF .STATUS THEN STATUS = .IO_STATUS[0];  
494 P 0914 2 IF NOT .STATUS THEN ERR_EXITT.STATUS);  
495 P 0915 2  
496 P 0916 2 ! If this is not a tape for interchange and the user has requested VMS  
497 P 0917 2 ! protection on the tape. Then write a VOL2 label after the VOL1 label.  
498 P 0918 2  
499 P 0919 2 IF NOT .INIT_OPTIONS[OPT_INTERCHG] AND .VMS_PROT NEQ 0  
500 P 0920 2 THEN  
501 P 0921 2 STATUS = $QIOW ( CHAN = .CHANNEL  
502 P 0922 2     IOSB = IO_STATUS[0],  
503 P 0923 2     FUNC = IOS_WRITEBLK,  
504 P 0924 2     P1 = VOL2,  
505 P 0925 2     P2 = 80);  
506 P 0926 2 IF .STATUS THEN STATUS = .IO_STATUS;  
507 P 0927 2 IF NOT .STATUS THEN ERR_EXITT.STATUS);  
508 P 0928 2  
509 P 0929 2  
510 P 0930 2 ! Give the user the opportunity to write the user volume labels, the first
```

511 0931 2 ! 3 characters of which must be 'UVL'. They should not be longer than 80 char-
512 0932 2 ! actors
513 0933 2
514 0934 2
515 0935 2 !STATUS = WRITE_USER_UVL();
516 0936 2 !IF NOT .STATUS THEN ERR_EXIT(.STATUS);
517 P 0937 2 STATUS = SQIOW(! HDR1
518 PPP 0938 2 CHAN = .CHANNEL
519 PPP 0939 2 IOSB = IO_STATUS[0],
520 PPP 0940 2 FUNC = IOS_WRITEBLK,
521 PPP 0941 2 P1 = HDR1,
522 PPP 0942 2 P2 = 80);
523 0943 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
524 0944 2 IF NOT .STATUS THEN ERR_EXIT(.STATUS);
525 0945 2
526 P 0946 2 STATUS = SQIOW(! HDR2
527 P 0947 2 CHAN = .CHANNEL
528 P 0948 2 IOSB = IO_STATUS[0],
529 P 0949 2 FUNC = IOS_WRITEBLK,
530 P 0950 2 P1 = HDR2,
531 P 0951 2 P2 = 80);
532 0952 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
533 0953 2 IF NOT .STATUS THEN ERR_EXIT(.STATUS);
534 0954 2
535 P 0955 2 STATUS = SQIOW(! Tape Mark
536 P 0956 2 CHAN = .CHANNEL
537 P 0957 2 IOSB = IO_STATUS[0],
538 P 0958 2 FUNC = IOS_WRITEEOF;
539 0959 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
540 0960 2 IF NOT .STATUS THEN ERR_EXIT(.STATUS);
541 0961 2
542 P 0962 2 STATUS = SQIOW(! Tape Mark
543 P 0963 2 CHAN = .CHANNEL
544 P 0964 2 IOSB = IO_STATUS[0],
545 P 0965 2 FUNC = IOS_WRITEEOF;
546 0966 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
547 0967 2 IF NOT .STATUS THEN ERR_EXIT(.STATUS);
548 0968 2
549 0969 2 HDR1[HD1SL_HD1LID] = 'EOF1'; ! format trailers
550 0970 2 HDR2[HD2SL_HD2LID] = 'EOF2';
551 0971 2
552 P 0972 2 STATUS = SQIOW(! EOF1
553 P 0973 2 CHAN = .CHANNEL
554 P 0974 2 IOSB = IO_STATUS[0],
555 P 0975 2 FUNC = IOS_WRITEBLK,
556 P 0976 2 P1 = HDR1,
557 P 0977 2 P2 = 80);
558 0978 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
559 0979 2 IF NOT .STATUS THEN ERR_EXIT(.STATUS);
560 P 0980 2 STATUS = SQIOW(! EOF2
561 P 0981 2 CHAN = .CHANNEL
562 P 0982 2 IOSB = IO_STATUS[0],
563 P 0983 2 FUNC = IOS_WRITEBLK,
564 P 0984 2 P1 = HDR2,
565 P 0985 2 P2 = 80);
566 0986 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
567 0987 2 IF NOT .STATUS THEN ERR_EXIT(.STATUS);

I 14
16-Sep-1984 01:50:56
14-Sep-1984 12:35:18VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[INIT.SRC]INITAP.B32;1Page 11
(2)

```

568 P 0988 2 STATUS = $QIOW(           ! Tape Mark
569 P 0989 2   CHAN = .CHANNEL,
570 P 0990 2   IOSB = IO_STATUS,
571 P 0991 2   FUNC = IOS_WRITEOF);
572 P 0992 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
573 P 0993 2 IF NOT .STATUS THEN ERR_EXITT.STATUS);
574 P 0994 2 STATUS = $QIOW(           ! Tape Mark
575 P 0995 2   CHAN = .CHANNEL,
576 P 0996 2   IOSB = IO_STATUS,
577 P 0997 2   FUNC = IOS_WRITEOF);
578 P 0998 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
579 P 0999 2 IF NOT .STATUS THEN ERR_EXITT.STATUS);
580 1000 2
581 P 1001 2 STATUS = $QIOW(
582 P 1002 2   CHAN = .CHANNEL,
583 P 1003 2   IOSB = IO_STATUS,
584 P 1004 2   FUNC = IOS_REWIND);
585 P 1005 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
586 P 1006 2 IF NOT .STATUS THEN ERR_EXITT.STATUS);
587 P 1007 2
588 P 1008 2 STATUS = $QIOW(
589 P 1009 2   CHAN = .CHANNEL,
590 P 1010 2   FUNC = IOS_AVAILABLE,
591 P 1011 2   IOSB = IO_STATUS[0]);
592 P 1012 2 RETURN 1;
593 1013 1 END;                         ! end of routine INIT_TAPE

```

00 00 41 31 31 45 4C 49 46 43 45 44 00000 P.AAA: .ASCII \DECFILE11A\<0><0>
 FFFFFFFF FF676980 0000C P.AAB: .LONG -10000000, -1

```

.TITLE INITAP
.IDENT \V04-000\
.PSECT SPLITS,NOWRT,NOEXE,2
.PSECT $0WN$,NOEXE,2

```

00000	ANSI_LABEL:	
	BLKB	80
00050	IO_STATUS:	
	BLKB	8
00058	PRIVILEGE_MASK:	
	BLKB	4
0005C	VOLUME_PROT:	
	BLKB	4
00060	VOLUME_UIC:	
	BLKB	4
00064	ACCESS:	
	BLKB	4
00068	CURRENT_RECORD:	
	BLKB	4
0006C	LABEL_VER:	
	BLKB	4
00070	UCB:	
	BLKB	4
00074	CHAR:	
	BLKB	4
00078	VOL1:	.ASCII \VOL1\
20	0007C	.ASCII \\

31 4C 4F 56
20

J 14
16-Sep-1984 01:50:56
14-Sep-1984 12:35:18VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[INIT.SRC]INITAP.B32;1Page 12
(2)

20	0007D	.ASCII	/
20	0007E	.ASCII	/
20	0007F	.ASCII	/
20	00080	.ASCII	/
20	00081	.ASCII	/
20	00082	.ASCII	/
20	00083	.ASCII	/
20	00084	.ASCII	/
20	00085	.ASCII	/
20	00086	.ASCII	/
20	00087	.ASCII	/
20	00088	.ASCII	/
20	00089	.ASCII	/
20	0008A	.ASCII	/
20	0008B	.ASCII	/
20	0008C	.ASCII	/
20	0008D	.ASCII	/
20	0008E	.ASCII	/
20	0008F	.ASCII	/
20	00090	.ASCII	/
20	00091	.ASCII	/
20	00092	.ASCII	/
20	00093	.ASCII	/
20	00094	.ASCII	/
20	00095	.ASCII	/
20	00096	.ASCII	/
20	00097	.ASCII	/
20	00098	.ASCII	/
20	00099	.ASCII	/
20	0009A	.ASCII	/
20	0009B	.ASCII	/
20	0009C	.ASCII	/
20	0009D	.ASCII	/
20	0009E	.ASCII	/
20	0009F	.ASCII	/
20	000A0	.ASCII	/
20	000A1	.ASCII	/
20	000A2	.ASCII	/
20	000A3	.ASCII	/
20	000A4	.ASCII	/
20	000A5	.ASCII	/
20	000A6	.ASCII	/
20	000A7	.ASCII	/
20	000A8	.ASCII	/
20	000A9	.ASCII	/
20	000AA	.ASCII	/
20	000AB	.ASCII	/
20	000AC	.ASCII	/
20	000AD	.ASCII	/
20	000AE	.ASCII	/
20	000AF	.ASCII	/
20	000B0	.ASCII	/
20	000B1	.ASCII	/
20	000B2	.ASCII	/
20	000B3	.ASCII	/
20	000B4	.ASCII	/
20	000B5	.ASCII	/

K 14
16-Sep-1984 01:50:56
14-Sep-1984 12:35:18VAX-11 Bliss-32 V4.0-742
DISKS\$VMSMASTER:[INIT.SRC]INITAP.B32;1Page 13
(2)32 4C 4F
43 25

20 000B6 .ASCII / /
000B7 .ASCII / /
000B8 .ASCII / /
000B9 .ASCII / /
000BA .ASCII / /
C30BB .ASCII / /
000BC .ASCII / /
000BD .ASCII / /
000BE .ASCII / /
000BF .ASCII / /
000C0 .ASCII / /
000C1 .ASCII / /
000C2 .ASCII / /
000C3 .ASCII / /
000C4 .ASCII / /
000C5 .ASCII / /
000C6 .ASCII / /
000C7 .ASCII /3/
56 000C8 VOL2:
44 000CC \VOL2\
20 000D0 \D%C\
000D1 .ASCII / /
000D2 .ASCII / /
000D3 .ASCII / /
000D4 .ASCII / /
000D5 .ASCII / /
000D6 .ASCII / /
000D7 .ASCII / /
000D8 .ASCII / /
000D9 .ASCII / /
000DA .ASCII / /
000DB .ASCII / /
000DC .ASCII / /
000DD .ASCII / /
000DE .ASCII / /
000DF .ASCII / /
000E0 .ASCII / /
000E1 .ASCII / /
000E2 .ASCII / /
000E3 .ASCII / /
000E4 .ASCII / /
000E5 .ASCII / /
000E6 .ASCII / /
000E7 .ASCII / /
000E8 .ASCII / /
000E9 .ASCII / /
000EA .ASCII / /
000EB .ASCII / /
000EC .ASCII / /
000ED .ASCII / /
000EE .ASCII / /
000EF .ASCII / /
000F0 .ASCII / /
000F1 .ASCII / /
000F2 .ASCII / /
20 000F3 .ASCII / /

L 14
16-Sep-1984 01:50:56
14-Sep-1984 12:35:18VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[INIT.SRC]INITAP.B32;1Page 14
(2)

31 52 44 48 HDR1:

20	000F4	.ASCII	/
20	000F5	.ASCII	/
20	000F6	.ASCII	/
20	000F7	.ASCII	/
20	000F8	.ASCII	/
20	000F9	.ASCII	/
20	000FA	.ASCII	/
20	000FB	.ASCII	/
20	000FC	.ASCII	/
20	000FD	.ASCII	/
20	000FE	.ASCII	/
20	000FF	.ASCII	/
20	00100	.ASCII	/
20	00101	.ASCII	/
20	00102	.ASCII	/
20	00103	.ASCII	/
20	00104	.ASCII	/
20	00105	.ASCII	/
20	00106	.ASCII	/
20	00107	.ASCII	/
20	00108	.ASCII	/
20	00109	.ASCII	/
20	0010A	.ASCII	/
20	0010B	.ASCII	/
20	0010C	.ASCII	/
20	0010D	.ASCII	/
20	0010E	.ASCII	/
20	0010F	.ASCII	/
20	00110	.ASCII	/
20	00111	.ASCII	/
20	00112	.ASCII	/
20	00113	.ASCII	/
20	00114	.ASCII	/
20	00115	.ASCII	/
20	00116	.ASCII	/
20	00117	.ASCII	\HDR1\
20	00118	.ASCII	
20	00119	.ASCII	
20	0011D	.ASCII	
20	0011E	.ASCII	
20	0011F	.ASCII	
20	00120	.ASCII	
20	00121	.ASCII	
20	00122	.ASCII	
20	00123	.ASCII	
20	00124	.ASCII	
20	00125	.ASCII	
20	00126	.ASCII	
20	00127	.ASCII	
20	00128	.ASCII	
20	00129	.ASCII	
20	0012A	.ASCII	
20	0012B	.ASCII	
20	0012C	.ASCII	
20	0012D	.ASCII	
20	0012E	.ASCII	
20	0012F	.ASCII	

M 14
16-Sep-1984 01:50:56
14-Sep-1984 12:35:18VAX-11 Bliss-32 V4.0-742
DISKS\$VMSMASTER:[INIT.SRC]INITAP.B32;1Page 15
(2)

20 20 41 31 31 45 40 49 46 43 45

30 30

32 52 44

20	00130	.ASCII	\ \
20	00131	.ASCII	\ \
20	00132	.ASCII	\ \
20	00133	.ASCII	\0\
20	00134	.ASCII	\0\
20	00135	.ASCII	\0\
20	00136	.ASCII	\0\
20	00137	.ASCII	\0\
20	00138	.ASCII	\0\
20	00139	.ASCII	\0\
20	0013A	.ASCII	\0\
20	0013B	.ASCII	\0\
20	0013C	.ASCII	\0\
20	0013D	.ASCII	\0\
20	0013E	.ASCII	\100\
20	00141	.ASCII	\ \
20	00142	.ASCII	\ \
20	00143	.ASCII	\ \
20	00144	.ASCII	\ \
20	00145	.ASCII	\ \
20	00146	.ASCII	\ \
20	00147	.ASCII	\ \
20	00148	.ASCII	\ \
20	00149	.ASCII	\ \
20	0014A	.ASCII	\ \
20	0014B	.ASCII	\ \
20	0014C	.ASCII	\ \
20	0014D	.ASCII	\ \
20	0014E	.ASCII	\0\
20	0014F	.ASCII	\0\
20	00150	.ASCII	\0\
20	00151	.ASCII	\0\
20	00152	.ASCII	\0\
20	00153	.ASCII	\0\
44	00154	.ASCII	\DECFILE11A \
20	00160	.ASCII	\ \
20	00161	.ASCII	\ \
20	00162	.ASCII	\ \
20	00163	.ASCII	\ \
20	00164	.ASCII	\ \
20	00165	.ASCII	\ \
20	00166	.ASCII	\ \
20	00167	.ASCII	\ \
48	00168	HDR2:	.ASCII \HDR2\
46	0016C	.ASCII	\F\
30	0016D	.ASCII	\0\
30	0016E	.ASCII	\0\
30	0016F	.ASCII	\0\
30	00170	.ASCII	\0\
30	00171	.ASCII	\0\
30	00172	.ASCII	\0\
30	00173	.ASCII	\0\
30	00174	.ASCII	\0\
30	00175	.ASCII	\0\
30	00176	.ASCII	\0\
30	00177	.ASCII	\0\
20	00178	.ASCII	\ \

N 14
16-Sep-1984 01:50:56
14-Sep-1984 12:35:18VAX-11 Bliss-32 V4.0-742
DISKS\$VMSMASTER:[INIT.SRC]INITAP.B32;1

Page 16 (2)

20	00179	.ASCII	/\
20	0017A	.ASCII	/\
20	0017B	.ASCII	/\
20	0017C	.ASCII	/\
20	0017D	.ASCII	/\
20	0017E	.ASCII	/\
20	0017F	.ASCII	/\
20	00180	.ASCII	/\
20	00181	.ASCII	/\
20	00182	.ASCII	/\
20	00183	.ASCII	/\
20	00184	.ASCII	/\
20	00185	.ASCII	/\
20	00186	.ASCII	/\
20	00187	.ASCII	/\
20	00188	.ASCII	/\
20	00189	.ASCII	/\
20	0018A	.ASCII	/\
20	0018B	.ASCII	/\
20	0018C	.ASCII	/\
20	0018D	.ASCII	/\
20	0018E	.ASCII	/\
20	0018F	.ASCII	/\
20	00190	.ASCII	/\
20	00191	.ASCII	/\
20	00192	.ASCII	/\
20	00193	.ASCII	/\
20	00194	.ASCII	/\
20	00195	.ASCII	/\
20	00196	.ASCII	/\
20	00197	.ASCII	/\
20	00198	.ASCII	/\
20	00199	.ASCII	/\
30	0019A	.ASCII	\00\
20	0019C	.ASCII	/\
20	0019D	.ASCII	/\
20	0019E	.ASCII	/\
20	0019F	.ASCII	/\
20	001A0	.ASCII	/\
20	001A1	.ASCII	/\
20	001A2	.ASCII	/\
20	001A3	.ASCII	/\
20	001A4	.ASCII	/\
20	001A5	.ASCII	/\
20	001A6	.ASCII	/\
20	001A7	.ASCII	/\
20	001A8	.ASCII	/\
20	001A9	.ASCII	/\
20	001AA	.ASCII	/\
20	001AB	.ASCII	/\
20	001AC	.ASCII	/\
20	001AD	.ASCII	/\
20	001AE	.ASCII	/\
20	001AF	.ASCII	/\
20	001B0	.ASCII	/\
20	001B1	.ASCII	/\
20	001B2	.ASCII	/\

20	001B3	.ASCII \ \
20	001B4	.ASCII \ \
20	001B5	.ASCII \ \
20	001B6	.ASCII \ \
20	001B7	.ASCII \ \

STARID= P.AAA
SECONDS= P.AAB

.EXTRN CALDAYNO, CONVDATE J2R
.EXTRN CONVDATE R2J, GET CHANNELUCB
.EXTRN GET RECORD, FORMAT VOLOWNER
.EXTRN LIB\$CVT OTB, PROCESS VOL2_LABEL
.EXTRN TAPE OWN PROT, CHANNEL
.EXTRN CTL\$GQ PROCPRIV
.EXTRN INIT OPTIONS, LABEL STRING
.EXTRN OWNER UIC, PROCESS DIC
.EXTRN PROTECTION, VOL ACC
.EXTRN VOL OWNER, ERASE BLOCKS
.EXTRN SYSSQIOW, SYSSSETIMR
.EXTRN SYSSWAITFR, SYSSCMKRL
.EXTRN SYSSASCTIM, SYSSMTACCESS
.EXTRN SYSSCMEXEC

.PSECT SCODES,NOWRT,2

OFFC 00000

SB 0000G CF 9E 00002
5A 00000000G 00 9E 00007
59 00000000G 00 9E 0000E
58 0000' CF 9E 00015
SE 14 C2 0001A

7E 7C 0001D
7E 7C 0001F
7E 7C 00021
7E 7C 00023
58 DD 00025
08 DD 00027
6B DD 00029
7E D4 0002B
OC FB 0002D
50 D0 00030
7E 7C 00033
7E 7C 00035
7E 7C 00037
7E 7C 00039
58 DD 0003B
24 DD 0003D
6B DD 0003F
7E D4 00041
OC FB 00043
50 D0 00046
52 D4 00049
7E 7C 0004B
7E 7C 0004D
7E 7C 0004F
7E 7C 00051

.ENTRY INIT TAPE. Save R2,R3,R4,R5,R6,R7,R8,R9,- : 0646
R10,R11

MOVAB CHANNEL, R11

MOVAB LIB\$STOP, R10

MOVAB SYSSQIOW, R9

MOVAB IO STATUS, R8

SUBL2 #20, SP

CLRQ -(SP)

CLRQ -(SP)

CLRQ -(SP)

CLRQ -(SP)

PUSHL R8

PUSHL #8

PUSHL CHANNEL

CLRL -(SP)

CALLS #12, SYSSQIOW

MOVL R0, STATUS

CLRQ -(SP)

CLRQ -(SP)

CLRQ -(SP)

PUSHL R8

PUSHL #36

PUSHL CHANNEL

CLRL -(SP)

CALLS #12, SYSSQIOW

MOVL R0, STATUS

CLRL J

CLRQ -(SP)

CLRQ -(SP)

CLRQ -(SP)

CLRQ -(SP)

69
56

69
56

18:

0749

0754

0758
0763

				PUSHL R8	
				PUSHL #8	
				PUSHL CHANNEL	
				CLRL -(SP)	
				CALLS #12, SYSSQIOW	
				MOVL R0, STATUS	
				CLRQ -(SP)	
				CLRQ -(SP)	
				CLRQ -(SP)	
				CLRQ -(SP)	
				PUSHL R8	
				PUSHL #36	
				PUSHL CHANNEL	
				CLRL -(SP)	
				CALLS #12, SYSSQIOW	
				MOVL R0, STATUS	
				BLBC STATUS, 2\$	
				MOVZWL IO STATUS, STATUS	
				CMPL STATUS, #420	
				BEQL 3\$	
				CMPL STATUS, #596	
				BNEQ 5\$	
				CLRQ -(SP)	
				PUSHAB SECONDS	
				CLRL -(SP)	
				CALLS #4, SYSSSETIMR	
				BLBC R0, 4\$	
				CLRL -(SP)	
				CALLS #1, SYSSWAITFR	
				AQBLEQ #9, J, 1\$	
				BLBS STATUS, 6\$	
				PUSHL STATUS	
				CALLS #1, LIBSTOP	
				CALLS #0, DEFAULT CHAR	
				MOVL #CTLSGQ_PROCPRI, PRIVILEGE_MASK	
				CHANNEL	
				#1	
				SP	
				PUSHAB GET_CHANNELUCB	
				CALLS #4, @SYSSCMKRNL	
				MOVL R0, UCB	
				BBC #3, INIT_OPTIONS+3, 7\$	
				BLBC #6, INIT_OPTIONS+3, 7\$	
				BLBC INIT_OPTIONS+5, 7\$	
				BBC #21, @PRIVILEGE_MASK, 7\$	
				BBS #18, @PRIVILEGE_MASK, 11\$	
				CALLS #0, READ VOLLABELS	
				ACCESS T1\$	
				BBC #3, INIT_OPTIONS+3, 8\$	
				BLBC INIT_OPTIONS+5, 9\$	
				BBS #21, @PRIVILEGE_MASK, 9\$	
				CMPL VOLUME_UIC, PROCESS_UIC	
				BNEQ 10\$	
				MOVO VOLUME_PROT, -(SP)	
				PUSHL #2	
				PUSHL SP	
				PUSHAB CHECK_PROT	

00000000G 9F 05 05 FB 00120 CALLS #5, @#SYSSCMKRNL
05 E8 00127 BLBS R0, 11S
24 DD 0012A 10\$: PUSHL #36
01 FB 0012C CALLS #1, LIB\$STOP
03 DO 0012F 11\$: MOVL #3, LABEL VER
00 FB 00133 CALLS #0, FORMAT VOL1_VOL2
50 DO 00138 MOVL R0, VMS PROT
08 DO 0013B MOVL #11, DESC
6E 9E 0013F MOVAB TODAY, DESC+4
7E 7C 00143 CLRQ -(SP)
AE 9F 00145 PUSHAB DESC
7E D4 00148 CLRL -(SP)
04 FB 0014A CALLS #4, SYSSASCTIM
C8 9F 00151 PUSHAB HDR1+41
AE 9F 00155 PUSHAB TODAY
02 FB 00158 CALLS #2, CONVDATE_R2J
06 28 0015D MOVC3 #6, HDR1+41, HDR1+47
A8 DD 00165 PUSHL UCB
01 DD 00168 PUSHL #1
5E DD 0016A PUSHL SP
CF 9F 0016C PUSHAB GET_RECORD
04 FB 00170 CALLS #4, @#SYSSCMKRNL
50 DO 00177 MOVL R0, CURRENT_RECORD
03 DD 0017B PUSHL #3
7E 7C 0017D CLRQ -(SP)
A8 DD 0017F PUSHL LABEL VER
CF DD 00182 PUSHL PROCESS_UIC
7E D4 00186 CLRL -(SP)
06 FB 00188 CALLS #6, SYSSMTACCESS
50 DO 0018F MOVL R0, CHAR
A8 DD 00193 PUSHL UCB
01 DD 00196 PUSHL #1
5E DD 00198 PUSHL SP
CF 9F 0019A PUSHAB GET_RECORD
04 FB 0019E CALLS #4, @#SYSSCMKRNL
56 56 18 50 DO 001A5 MOVL R0, STATUS
56 18 A8 D1 001A8 CMPL CURRENT_RECORD, STATUS
08 13 001AC BEQL 12S
7E 0224 8F 3C 001AE MOVZWL #548, -(SP)
6A 2C 00FD C8 01 FB 001B3 CALLS #1, LIB\$STOP
24 A8 90 001B6 12\$: MOVB CHAR, HDR1+53
06 28 001BC MOVC3 #6, VOL1+4, HDR1+21
7E 7C 001C3 CLRQ -(SP)
7E 7C 001C5 CLRQ -(SP)
7E 7C 001C7 CLRQ -(SP)
7E 7C 001C9 CLRQ -(SP)
58 DD 001CB PUSHL R8
24 DD 001CD PUSHL #36
6B DD 001CF PUSHL CHANNEL
7E D4 001D1 CLRL -(SP)
69 OC FB 001D3 CALLS #12, SYSSQIOW
56 56 06 50 DO 001D6 MOVL R0, STATUS
56 68 F9 001D9 BLBC STATUS, 13S
05 56 68 3C 001DC MOVZWL IO_STATUS, STATUS
56 56 EB 001DF BLBS STATUS, 14S
56 6A DD 001E2 13\$: PUSHL STATUS
01 FB 001E4 CALLS #1, LIB\$STOP

31	0000V	05	0000G	CF	E9	001E7	14\$:	BLBC CALLS BBC PUSHL PUSHL MOVQ PUSHL PUSHAB	INIT_OPTIONS, 15\$ #0, SET DENSITY #2, INIT_OPTIONS+5, 17\$ CHANNEL #1 #3, -(SP) SP	0882
	0000G	CF		02	E1	001F1	15\$:	PUSHL MOVQ PUSHL PUSHAB	#1 #3, -(SP)	0892
		7E		6B	DD	001F7		MOVZWL	SP	0895
				01	DD	001F9		BLBS	ERASE BLOCKS	
				03	7D	001FB		PUSHL	#6, BSSYSSCMEXEC	
				5E	DD	001FE		CALLS	RO, STATUS	
	00000000G	9F	0000G	CF	9F	00200		MOVL	STATUS	0897
		56		06	FB	00204		BLBC	STATUS, 16\$	0898
		06		56	DO	0020B		MOVZWL	IO STATUS, STATUS	0900
		56		6B	E9	0020E		BLBS	STATUS, 17\$	
		11		56	3C	00211		PUSHL	STATUS	
				56	E8	00214		CLRL	-(SP)	
				56	DD	00217	16\$:	PUSHL	#7704592	
				7E	D4	00219		CALLS	#3, LIB\$SIGNAL	
	00000000G	00	00759010	8F	DD	0021B		CLRQ	-(SP)	0912
				03	FB	00221		CLRQ	-(SP)	
		7E	50	7E	7C	00228	17\$:	MOVZBL	#80, -(SP)	
			28	8F	9A	0022C		PUSHAB	VOL1	
				A8	9F	00230		CLRQ	-(SP)	
				7E	7C	00233		PUSHL	R8	
				58	DD	00235		PUSHL	#32	
				20	DD	00237		PUSHL	CHANNEL	
				6B	DD	00239		CLRL	-(SP)	
				7E	D4	0023B		CALLS	#12, SYSSQIOW	
				69	OC	0023D		MOVL	RO, STATUS	
				56	DO	00240		BLBC	STATUS, 18\$	0913
				06	E9	00243		MOVZWL	IO STATUS, STATUS	0914
				56	3C	00246		BLBS	STATUS, 19\$	
				05	E8	00249		PUSHL	STATUS	
		6A	0000G	01	FB	0024E	18\$:	CALLS	#1, LIB\$STOP	
				57	E0	00251	19\$:	BBS	#1, INIT_OPTIONS+5, 20\$	0919
				1B	D5	00257		TSTL	VM\$_PROT	
				7E	13	00259		BEQL	20\$	
				7E	7C	0025B		CLRQ	-(SP)	0925
		7E	50	7E	7C	0025D		CLRQ	-(SP)	
			78	8F	9A	0025F		MOVZBL	#80, -(SP)	
				A8	9F	00263		PUSHAB	VOL2	
				7E	7C	00266		CLRQ	-(SP)	
				58	DD	00268		PUSHL	R8	
				20	DD	0026A		PUSHL	#32	
				6B	DD	0026C		PUSHL	CHANNEL	
				7E	D4	0026E		CLRL	-(SP)	
				69	OC	00270		CALLS	#12, SYSSQIOW	
				56	DO	00273		MOVL	RO, STATUS	
				06	E9	00276	20\$:	BLBC	STATUS, 21\$	0926
				56	DO	00279		MOVL	IO STATUS, STATUS	0927
				6B	DO	0027C		BLBS	STATUS, 22\$	
				05	E8	0027F	21\$:	PUSHL	STATUS	
		6A		56	DD	0027F		CALLS	#1, LIB\$STOP	
				01	FB	00281		CLRQ	-(SP)	
				7E	7C	00284	22\$:	CLRQ	-(SP)	
		7E	50	7E	7C	00286		MOVZBL	#80, -(SP)	0942
			00C8	8F	9A	00288		PUSHAB	HDR1	
				C8	9F	0028C		CLRQ	-(SP)	
				7E	7C	00290				

			PUSHL R8	
			PUSHL #32	
			PUSHL CHANNEL	
			CLRL -(SP)	
			CALLS #12, SYSSQIOW	
			MOVL R0, STATUS	
			BLBC STATUS, 23\$	
			MOVZWL IO STATUS, STATUS	
			BLBS STATUS, 24\$	
			23\$: PUSHL STATUS	0943
			CALLS #1, LIB\$STOP	
			CLRQ -(SP)	
			CLRQ -(SP)	
			MOVZBL #80, -(SP)	
			PUSHAB HDR2	
			CLRQ -(SP)	
			PUSHL R8	
			PUSHL #32	
			PUSHL CHANNEL	
			CLRL -(SP)	
			CALLS #12, SYSSQIOW	
			MOVL R0, STATUS	
			BLBC STATUS, 25\$	
			MOVZWL IO STATUS, STATUS	0952
			BLBS STATUS, 26\$	
			25\$: PUSHL STATUS	0953
			CALLS #1, LIB\$STOP	
			CLRQ -(SP)	
			PUSHL R8	
			PUSHL #40	
			PUSHL CHANNEL	
			CLRL -(SP)	
			CALLS #12, SYSSQIOW	
			MOVL R0, STATUS	
			BLBC STATUS, 27\$	
			MOVZWL IO STATUS, STATUS	0959
			BLBS STATUS, 28\$	
			27\$: PUSHL STATUS	0960
			CALLS #1, LIB\$STOP	
			CLRQ -(SP)	
			PUSHL R8	
			PUSHL #40	
			PUSHL CHANNEL	
			CLRL -(SP)	
			CALLS #12, SYSSQIOW	
			MOVL R0, STATUS	
			BLBC STATUS, 29\$	
			MOVZWL IO STATUS, STATUS	0966
			BLBS STATUS, 30\$	
			29\$: PUSHL STATUS	0967
			CALLS #1, LIB\$STOP	

00C8	C8 31464F45	8F DD 00320	30\$:	MOVL #826691397, HDR1	0969
0118	C8 32464F45	8F DD 00329		MOVL #843468613, HDR2	0970
		7E 7C 00332		CLRQ -(SP)	0977
		7E 7C 00334		CLRQ -(SP)	
7E	50	8F 9A 00336		MOVZBL #80 -(SP)	
		C8 9F 0033A		PUSHAB HDR1	
		7E 7C 0033E		CLRQ -(SP)	
		58 DD 00340		PUSHL R8	
		20 DD 00342		PUSHL #32	
		6B DD 00344		PUSHL CHANNEL	
		7E D4 00346		CLRL -(SP)	
69		OC FB 00348		CALLS #12, SYS\$QIOW	
56		50 DO 0034B		MOVL R0, STATUS	0978
06		56 E9 0034E		BLBC STATUS, 31\$	
56		68 3C 00351		MOVZWL IO STATUS, STATUS	
05		56 E8 00354		BLBS STATUS, 32\$	0979
6A		56 DD 00357	31\$:	PUSHL STATUS	
		01 FB 00359		CALLS #1, LIB\$STOP	
		7E 7C 0035C	32\$:	CLRQ -(SP)	0985
7E	50	8F 9A 00360		MOVZBL #80, -(SP)	
		C8 9F 00364		PUSHAB HDR2	
		7E 7C 00368		CLRQ -(SP)	
		58 DD 0036A		PUSHL R8	
		20 DD 0036C		PUSHL #32	
		6B DD 0036E		PUSHL CHANNEL	
		7E D4 00370		CLRL -(SP)	
69		OC FB 00372		CALLS #12, SYS\$QIOW	
56		50 DO 00375		MOVL R0, STATUS	
06		56 E9 00378		BLBC STATUS, 33\$	0986
56		68 3C 0037B		MOVZWL IO STATUS, STATUS	
05		56 E8 0037E		BLBS STATUS, 34\$	0987
6A		56 DD 00381	33\$:	PUSHL STATUS	
		01 FB 00383		CALLS #1, LIB\$STOP	
		7E 7C 00386	34\$:	CLRQ -(SP)	0991
		7E 7C 00388		CLRQ -(SP)	
		7E 7C 0038A		CLRQ -(SP)	
		7E 7C 0038C		CLRQ -(SP)	
		58 DD 0038E		PUSHL R8	
		28 DD 00390		PUSHL #40	
		6B DD 00392		PUSHL CHANNEL	
		7E D4 00394		CLRL -(SP)	
69		OC FB 00396		CALLS #12, SYS\$QIOW	
56		50 DO 00399		MOVL R0, STATUS	
06		56 E9 0039C		BLBC STATUS, 35\$	0992
56		68 3C 0039F		MOVZWL IO STATUS, STATUS	
05		56 E8 003A2		BLBS STATUS, 36\$	0993
6A		56 DD 003A5	35\$:	PUSHL STATUS	
		01 FB 003A7		CALLS #1, LIB\$STOP	
		7E 7C 003AA	36\$:	CLRQ -(SP)	0997
		7E 7C 003AC		CLRQ -(SP)	
		7E 7C 003AE		CLRQ -(SP)	
		7E 7C 003B0		CLRQ -(SP)	
		58 DD 003B2		PUSHL R8	
		28 DD 003B4		PUSHL #40	
		6B DD 003B6		PUSHL CHANNEL	
		7E D4 003B8		CLRL -(SP)	

69	0C	FB	003BA	CALLS	#12, SYSSQIOW	
56	50	DO	003BD	MOVL	R0, STATUS	0998
06	56	E9	003C0	BLBC	STATUS, 37\$	
56	68	3C	003C3	MOVZWL	IO STATUS, STATUS	
05	56	E8	003C6	BLBS	STATUS, 38\$	0999
6A	56	DD	003C9	37\$: PUSHL	STATUS	
	01	FB	003CB	CALLS	#1, LIB\$STOP	1004
	7E	7C	003CE	CLRQ	-(SP)	
	7E	7C	003D0	CLRQ	-(SP)	
	7E	7C	003D2	CLRQ	-(SP)	
	7E	7C	003D4	CLRQ	-(SP)	
	58	DD	003D6	PUSHL	R8	
	24	DD	003D8	PUSHL	#36	
	6B	DD	003DA	PUSHL	CHANNEL	
	7E	D4	003DC	CLRL	-(SP)	
69	0C	FB	003DE	CALLS	#12, SYSSQIOW	
56	50	DO	003E1	MOVL	R0, STATUS	1005
06	56	E9	003E4	BLBC	STATUS, 39\$	
56	68	3C	003E7	MOVZWL	IO STATUS, STATUS	
05	56	E8	003EA	BLBS	STATUS, 40\$	1006
6A	56	DD	003ED	39\$: PUSHL	STATUS	
	01	FB	003EF	CALLS	#1, LIB\$STOP	1011
	7E	7C	003F2	CLRQ	-(SP)	
	7E	7C	003F4	CLRQ	-(SP)	
	7E	7C	003F6	CLRQ	-(SP)	
	7E	7C	003F8	CLRQ	-(SP)	
	58	DD	003FA	PUSHL	R8	
	11	DD	003FC	PUSHL	#17	
	6B	DD	003FE	PUSHL	CHANNEL	
	7E	D4	00400	CLRL	-(SP)	
69	0C	FB	00402	CALLS	#12, SYSSQIOW	
56	50	DO	00405	MOVL	R0, STATUS	1013
	04	00408	RET			

; Routine Size: 1033 bytes. Routine Base: SCODES + 0000

```
595    1014 1 ROUTINE DEFAULT_CHAR : NOVALUE =
596    1015 1
597    1016 1 ++
598    1017 1
599    1018 1 FUNCTIONAL DESCRIPTION:
600    1019 1
601    1020 1      This routine sets the tape drive default characteristics.
602    1021 1
603    1022 1 CALLING SEQUENCE:
604    1023 1      DEFAULT_CHAR ();
605    1024 1
606    1025 1 INPUT PARAMETERS:
607    1026 1      NONE
608    1027 1
609    1028 1 IMPLICIT INPUTS:
610    1029 1      CHANNEL      - the I/O channel of the tape drive
611    1030 1
612    1031 1 OUTPUT PARAMETERS:
613    1032 1      NONE
614    1033 1
615    1034 1 IMPLICIT OUTPUTS:
616    1035 1      IO_STATUS     - set to the return status of the QIO
617    1036 1
618    1037 1 ROUTINE VALUE:
619    1038 1      NONE
620    1039 1
621    1040 1 SIDE EFFECTS:
622    1041 1      NONE
623    1042 1
624    1043 1 USER ERRORS:
625    1044 1      NONE
626    1045 1
627    1046 1      --
628    1047 1
629    1048 2 BEGIN
630    1049 2
631    1050 2 LITERAL
632    1051 2      ODD_PARITY = 0;
633    1052 2
634    1053 2 LOCAL
635    1054 2      CHARACTERISTIC : VECTOR [4,WORD], ! characteristics to set
636    1055 2      STATUS;
637    1056 2
638    1057 2 BIND
639    1058 2      ! Set up offsets into the characteristics buffer
640    1059 2
641    1060 2      FORMAT      = CHARACTERISTIC[2] : BBLOCK,
642    1061 2      PARITY      = CHARACTERISTIC[2] : BBLOCK,
643    1062 2      BUFFER_SIZE = CHARACTERISTIC[1] : WORD,
644    1063 2      DENSITY     = CHARACTERISTIC[2] : BBLOCK;
645    1064 2
646    1065 2      CHARACTERISTIC[0]=CHARACTERISTIC[1]=CHARACTERISTIC[2]=CHARACTERISTIC[3]=0;
647    1066 2
648    1067 2      ! Now set density
649    1068 2
650    1069 2      DENSITY[MTSV_DENSITY] = MTSK_PE_1600;
651    1070 2
```

```

652 1071 2 | Parity set to odd, we only support 9-tracks and 9-tracks are always odd
653 1072 2
654 1073 2 | PARITY [ MTSV_PARITY ] = ODD_PARITY;
655 1074 2
656 1075 2 | Reset Tape format to FILES-11 ( only supported format )
657 1076 2
658 1077 2 | FORMAT [ MTSV_FORMAT ] = MTSK_NORMAL11;
659 1078 2
660 1079 2 | Set the buffer size to ANSI max ( VMS default )
661 1080 2
662 1081 2 | BUFFER_SIZE = 2048;
663 1082 2
664 1083 2 | write the characteristics to the tape drive
665 1084 2
666 P 1085 2 STATUS = $QIOW (CHAN = .CHANNEL,
667 P 1086 2 IOSB = IO_STATUS,
668 P 1087 2 FUNC = IOS_SETMODE,
669 P 1088 2 P1 = CHARACTERISTIC);
670 1089 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
671 1090 2 IF NOT .STATUS THEN ERR_EXIT(.STATUS);
672 1091 2
673 1092 1 END;                                ! end of routine DEFAULT_CHAR

```

0000 00000 DEFAULT_CHAR:								
05 AE	05	00	7E	7C 00002	.WORD	Save nothing		1014
04 AE	04	AE	04	F0 00004	CLRQ	CHARACTERISTIC		1065
	02	AE	0800	08 8A 0000A	INSV	#4, #0, #5, DENSITY+1		1069
				0C F0 0000E	BICB2	#8, PARITY		1073
				8F B0 00014	INSV	#12, #4, #4, FORMAT		1077
				7E 7C 0001A	MOVW	#2048, BUFFER_SIZE		1081
				7E 7C 0001C	CLRQ	-(SP)		1088
				7E D4 0001E	CLRQ	-(SP)		
			14	AE 9F 00020	CLRL	-(SP)		
				7E 7C 00023	PUSHAB	CHARACTERISTIC		
				CF 9F 00025	CLRQ	-(SP)		
				23 DD 00029	PUSHAB	IO_STATUS		
				CF DD 0002B	PUSHL	#35		
				7E D4 0002F	PUSHL	CHANNEL		
			00000000G	OC FB 00031	CLRL	-(SP)		
			00	50 F9 00038	CALLS	#12, SYSSQIOW		
			08	CF 3C 0003B	BLBC	STATUS, 1\$		1089
			50	50 E8 00040	MOVZWL	IO_STATUS, STATUS		
			09	50 DD 00043	BLBS	STATUS, 2\$		1090
				01 FB 00045	PUSHL	STATUS		
			00000000G	04 0004C	CALLS	#1, LIB\$STOP		
			00	28:	RET			1092

: Routine Size: 77 bytes. Routine Base: SCODE\$ + 0409

```
675    1093 1 ROUTINE SET_DENSITY : NOVALUE =
676    1094 1
677    1095 1 !++
678    1096 1
679    1097 1 FUNCTIONAL DESCRIPTION:
680    1098 1
681    1099 1 This routine sets the density of the tape drive.
682    1100 1
683    1101 1 CALLING SEQUENCE:
684    1102 1     SET_DENSITY ();
685    1103 1
686    1104 1 INPUT PARAMETERS:
687    1105 1     NONE
688    1106 1
689    1107 1 IMPLICIT INPUTS:
690    1108 1     CHANNEL      - the I/O channel of the tape drive
691    1109 1
692    1110 1 OUTPUT PARAMETERS:
693    1111 1     NONE
694    1112 1
695    1113 1 IMPLICIT OUTPUTS:
696    1114 1     IO_STATUS   - set to the return status of the QIO
697    1115 1
698    1116 1
699    1117 1 ROUTINE VALUE:
700    1118 1     NONE
701    1119 1
702    1120 1
703    1121 1
704    1122 1
705    1123 1
706    1124 1
707    1125 1
708    1126 1
709    1127 2 BEGIN
710    1128 2
711    1129 2 LOCAL CHARACTERISTIC : VECTOR [4,WORD],      ! characteristics to set
712    1130 2
713    1131 2 STATUS:
714    1132 2
715    1133 2
716    1134 2
717    1135 2
718    1136 2
719    1137 2 BIND      ! Set up offsets into the characteristics buffer
720    1138 2
721    1139 2
722    1140 2
723    1141 2
724    P 1142 2 STATUS = $QIOW (CHAN = .CHANNEL,
725          1143 2           IOSB = CHARACTERISTIC,
726          1144 2           FUNC = IOS_SENSEMODE);
727    1145 2 IF .STATUS THEN STATUS = .CHARACTERISTIC[0];
728    1146 2 IF NOT .STATUS THEN ERR_EXIT (.STATUS);
729    1147 2
730    1148 2
731    1149 2 ! Set up the buffer to hold the new characteristics. Get the device
           1149 2 independent stuff from the 2nd long word of IO_STATUS, use the default
```

001C 00000 SET_DENSITY:

	7E	7C	00063	5\$:	CLRQ	-(SP)	1169
	7E	7C	00065		CLRQ	-(SP)	
	7E	D4	00067		CLRL	-(SP)	
14	AE	9F	00069		PUSHAB	CHARACTERISTIC	
	7E	7C	0006C		CLRQ	-(SP)	
0000'	CF	9F	0006E		PUSHAB	IO STATUS	
	23	DD	00072		PUSHL	#35	
0000G	CF	DD	00074		PUSHL	CHANNEL	
	7E	D4	00078		CLRL	-(SP)	
63	0C	FB	0007A		CALLS	#12, SYSSQIOW	
52	50	DD	0007D		MOVL	RO STATUS	
08	52	F9	00080		BLBC	STATUS 6\$	
52	CF	3C	00083		MOVZWL	IO STATUS STATUS	
05	52	EB	00088		BLBS	STATUS, 7\$	
64	52	DD	0008B	6\$:	PUSHL	STATUS	
	01	FB	0008D		CALLS	#1, LIB\$STOP	
	04	00090	7\$:		RET		1173

; Routine Size: 145 bytes, Routine Base: SCODES + 0456

757 1174 1 ROUTINE READ_VOLLABELS : NOVALUE =
758 1175 1
759 1176 1 !++
760 1177 1
761 1178 1 FUNCTIONAL DESCRIPTION:
762 1179 1 this routine reads the first block on the magnetic tape and
763 1180 1 checks if it is an ANSI tape. If it is, it then reads the
764 1181 1 HDR1 record to determine if the first file on the tape has expired.
765 1182 1
766 1183 1 CALLING SEQUENCE:
767 1184 1 READ_VOLLABELS()
768 1185 1
769 1186 1 INPUT PARAMETERS:
770 1187 1 none
771 1188 1
772 1189 1 IMPLICIT INPUTS:
773 1190 1 channel - channel number assigned to device being initialized
774 1191 1
775 1192 1 OUTPUT PARAMETERS:
776 1193 1 none
777 1194 1
778 1195 1 IMPLICIT OUTPUTS:
779 1196 1 VOLUME_UIC - owner of tape
780 1197 1 VOLUME_PROT - tape protection
781 1198 1 ACCESS - users' access to a magnetic tape volume is set
782 1199 1
783 1200 1 ROUTINE VALUE:
784 1201 1 none
785 1202 1
786 1203 1 SIDE EFFECTS:
787 1204 1 none
788 1205 1
789 1206 1 USER ERRORS:
790 1207 1 none
791 1208 1
792 1209 1 --
793 1210 1
794 1211 2 BEGIN
795 1212 2
796 1213 2 LOCAL
797 1214 2 DATE : VECTOR [2], : binary date
798 1215 2 DESCRIPTOR : VECTOR [2], : descriptor for today buffer
799 1216 2 REGDATE : VECTOR [12,BYTE], : buffer for date in format
800 1217 2 DD MMM YYYY
801 1218 2 STATUS, : system service status
802 1219 2 TODAY : VECTOR [12,BYTE],
803 1220 2 VMS_TAPE : BITVECTOR [1]; : set if the VOL1 sys code is VMS
804 1221 2
805 1222 2 ! read first block on tape and check status
806 1223 2
807 P 1224 2 STATUS = SQIOW(
808 P 1225 2 CHAN = .CHANNEL,
809 P 1226 2 FUNC = IOS_READLBLK,
810 P 1227 2 IOSB = IO_STATUS,
811 P 1228 2 P1 = ANSI_LABEL,
812 P 1229 2 P2 = 80);
813 1230 2 IF .STATUS THEN \$STATUS = .IO_STATUS[0];

```
814 1231 2
815 1232 2 | set up default volume owner and protection, which is the current users UIC
816 1233 2 | and read/write allowed. This will be reset by TAPE_OWNER_PROT if this is
817 1234 2 | a VAX/VMS tape
818 1235 2
819 1236 2 VOLUME_UIC = .PROCESS_UIC;
820 1237 2 VOLUME_PROT = 0;
821 1238 2
822 1239 2
823 1240 2 | if first record is Tape Mark then not ANSI tape
824 1241 2 | if label is more than 80 characters ignore those characters beyond 80
825 1242 2
826 1243 2 IF (.NOT .STATUS) AND (.STATUS NEQ SSS_DATAOVERUN)
827 1244 2 THEN
828 1245 2 BEGIN
829 1246 2
830 1247 3 ! if this is a new tape, the default density may have been changed
831 1248 3 ! by the QIO failure
832 1249 3
833 1250 3
834 1251 3 IF .STATUS EQL SSS_OPINCOMPL
835 1252 3 THEN
836 1253 4 BEGIN
837 1254 4
838 1255 4 ! tape must be at beginning ( no reads to set density )
839 P 1256 4
840 P 1257 4 STATUS = $QIOW( CHAN = .CHANNEL,
841 1258 4 FUNC = IOSREWIND,
842 1259 4 IOSB = IO_STATUS);
843 1260 4 IF .STATUS THEN STATUS = .IO_STATUS[0];
844 1261 4 IF NOT .STATUS THEN ERR_EXIT(.STATUS);
845 1262 4 DEFAULT_CHAR();
846 1263 3
847 1264 3
848 1265 3 RETURN 1;
849 1266 2
850 1267 2
851 1268 2 ! now check if first block is VOL1, foreign
852 1269 2
853 1270 2 IF .ANSI_LABEL[VL1$L_VL1LID] NEQ 'VOL1' THEN RETURN 1;
854 1271 2
855 1272 2 ! Get the ANSI standard version off the tape.
856 1273 2
857 1274 2 LABEL_VER = .ANSI_LABEL[VL1$B_LBLSTDVER] - '0';
858 1275 2
859 1276 2 ! Call the accessibility system service to check the accessibility char
860 1277 2 | on the VOL1 label.
861 1278 2 | First keep the record that the UCB is reading. The accessibility
862 1279 2 | routine can not move the tape from under us! Thus we will compare
863 1280 2 | this to the field after the call and if the tape was moved we punt
864 1281 2 | the operation.
865 1282 2
866 1283 2 CURRENT_RECORD = KERNEL_CALL(GET_RECORD,.UCB);
867 1284 2
868 P 1285 2 ACCESS = $MTACCESS(LBLNAM = ANSI_LABEL,
869 1286 2 UIC = .PROCESS_UIC,
870 P 1287 2 STD_VERSION = .LABEL_VER.
```

```
871 P 1288 2
872 P 1289 2
873 P 1290 2
874 P 1291 2
875 P 1292 2
876 P 1293 2
877 P 1294 2
878 P 1295 2
879 P 1296 2
880 P 1297 2
881 P 1298 2
882 P 1299 2
883 P 1300 2
884 P 1301 2
885 P 1302 2
886 P 1303 2
887 P 1304 2
888 P 1305 2
889 P 1306 2
890 P 1307 2
891 P 1308 2
892 P 1309 2
893 P 1310 3
894 P 1311 3
895 P 1312 3
896 P 1313 3
897 P 1314 3
898 P 1315 3
899 P 1316 2
900 P 1317 2
901 P 1318 2
902 P 1319 2
903 P 1320 2
904 P 1321 2
905 P 1322 2
906 P 1323 2
907 P 1324 2
908 P 1325 2
909 P 1326 2
910 P 1327 2
911 P 1328 2
912 P 1329 2
913 P 1330 2
914 P 1331 2
915 P 1332 2
916 P 1333 2
917 P 1334 2
918 P 1335 2
919 P 1336 2
920 P 1337 2
921 P 1338 2
922 P 1339 2
923 P 1340 2
924 P 1341 2
925 P 1342 2
926 P 1343 2
927 P 1344 2

        ACCESS_CHAR = 0,
        ACCESS_SPEC = MTASK_NOCHAR,
        TYPE = MTASK_INVOL1;

STATUS = KERNEL CALL(GET_RECORD,.UCB);
IF .CURRENT_RECORD NEQ .STATUS
    THEN ERR_EXIT(SSS_TAPEPOSLOST);

! Now check the ACCESS returned from the service. For SSS_FILACCERR
! check to make sure / OVERRIDE=ACCESS was specified and the user
! has privilege then set to check VMS protection.
! For SSS_NOFILACC, SSS_NOVOLACC return the code
! to the user. In this case the user has no access to the tape volume.
! For a 0 give the user all access. For SSS_NORMAL check the VMS
! protection.

IF .ACCESS EQL SSS_NOVOLACC
    OR .ACCESS EQL SSS_NOFILACC
    THEN ERR_EXIT(.ACCESS);

IF .ACCESS EQL SSS_FILACCERR
    THEN
        BEGIN
            IF NOT .INIT OPTIONS[OPT_OVR_ACC]
                THEN ERR_EXIT(.ACCESS);
            IF NOT .PRIVILEGE MASK[PRV$V_VOLPRO]
                THEN ERR_EXIT(.ACCESS);
            ACCESS = SSS_NORMAL;
        END;

! Determine owner and VMS protection of the tape. If not VMS protected
! and pre ANSI version 4 and a DEC operating system wrote the tape
! then the user must override the owner id field.

STATUS = TAPE OWN PROT(VOLUME_UIC, VOLUME_PROT, .PROCESS_UIC, ANSI_LABEL);
! If ACCESS allows see if user has VMS privilege to init the volume.
! Also set the VOLUME_PROT accordingly.

IF .ACCESS
    THEN
        BEGIN
            IF NOT .STATUS AND NOT .INIT_OPTIONS[OPT_OVR_VOL0]
                THEN ERR_EXIT(SSS_VOL0ERR);
        END
    ELSE
        VOLUME_PROT = 0;

! check to see if the VOL1 system code is VMS's if it isn't then we don't
! process the VOL2 label.

IF CHSEQL(10,STARID,10,ANSI_LABEL[VL1$T_SYS_CODE],0)
    THEN VMS_TAPE = 1
    ELSE VMS_TAPE = 0;

! first record on tape is VOL1. Now read HDR1 and determine if first
```

```
928 1345 2 ! file has expired. User volume labels may intervene.  
929 1346 2  
930 1347 2 WHILE 1 DO  
931 1348 3 BEGIN  
932 P 1349 3 STATUS = SQIOW(  
933 P 1350 3     CHAN = .CHANNEL,  
934 P 1351 3     FUNC = IOS READBLK,  
935 P 1352 3     IO$B = IO_STATUS[0],  
936 P 1353 3     P1 = ANSI_LABEL,  
937 1354 3     P2 = BO);  
938 1355 3 IF .STATUS THEN STATUS = .IO_STATUS[0];  
939 1356 3 IF NOT .STATUS THEN  
940 1357 3     IF .STATUS NEQ SSS_DATAOVERUN THEN RETURN 0; ! ANSI tape, but can't  
941 1358 3             ! read HDR1  
942 1359 3             ! If the sys code of the VOL1 label indicates that this is a VMS tape  
943 1360 3             ! and we find a VOL2 label then process the label.  
944 1361 3  
945 1362 3 IF .VMS_TAPE AND .ANSI_LABEL[VL2$L_VL2LID] EQL 'VOL2'  
946 1363 3 THEN  
947 1364 4 BEGIN  
948 1365 4     PROCESS_VOL2_LABEL (VOLUME_UIC, VOLUME_PROT, .PROCESS_UIC,  
949 1366 4             ANSI_LABEL);  
950 1367 4             IF NOT .ACCESS THEN VOLUME_PROT= 0;  
951 1368 3 END:  
952 1369 3 IF .ANSI_LABEL[HD1$L_HD1LID] EQL 'HDR1' THEN EXITLOOP;  
953 1370 2 END;  
954 1371 2  
955 1372 2 ! test if the first file on the tape has expired  
956 1373 2 ! convert the JULIAN date on the tape to a VMS date  
957 1374 2  
958 1375 2 IF CONVDATE_J2R(REGDATE,ANSI_LABEL[HD1$T_EXPIREDT])  
959 1376 2 THEN  
960 1377 3 BEGIN  
961 1378 3     DESCRIPTOR[0] = 12; ! set up the descriptor  
962 1379 3     DESCRIPTOR[1] = REGDATE;  
963 1380 3     REGDATE[11] = ',';  
964 1381 3     SBINTIM(TIMBUF = DESCRIPTOR,TIMADR = DATE); ! convert from ASCII to binary  
965 1382 3     SGETTIM(TIMADR = TODAY); ! get today's date in binary  
966 1383 3     CALDAYNO(DATE,TODAY); ! chop off hours min and sec  
967 1384 3 END  
968 1385 2 ELSE DATE = TODAY = 0; ! when all else fails  
969 1386 2  
970 1387 3 IF (.DATE GTRU .TODAY) AND NOT (.INIT_OPTIONS[OPT_OVR_EXP])  
971 1388 3     THEN ERR_EXIT (SSS_FILENOEXP);  
972 1389 2  
973 1390 2 ! Call the accessibility system service to check the accessibility char  
974 1391 2 on the HDR1 label.  
975 1392 2 First keep the record that the UCB is reading. The accessibility  
976 1393 2 routine can not move the tape from under us! Thus we will compare  
977 1394 2 this to the field after the call and if the tape was moved we punt  
978 1395 2 the operation.  
979 1396 2  
980 1397 2 CURRENT_RECORD = KERNEL_CALL(GET_RECORD,.UCB);  
981 1398 2  
P 1399 2 ACCESS = SMTACCESS(LBLNAM = ANSI_LABEL,  
P 1400 2             UIC = .PROCESS_UIC,  
P 1401 2             STD_VERSION = :LABEL_VER,
```

```

985 P 1402 2 ACCESS_CHAR = 0
986 P 1403 2 ACCESS_SPEC = MTASK_NOCHAR,
987 1404 2 TYPE = MTASK_INHDR1;
988 1405 2
989 1406 2 STATUS = KERNEL CALL(GET_RECORD,.UCB);
990 1407 2 IF .CURRENT_RECORD NEQ .STATUS
991 1408 2 THEN ERR_EXIT(SSS_TAPEPOSLOST);
992 1409 2
993 1410 2 ! Now check the ACCESS returned from the service. For SSS_FILACCERR
994 1411 2 check to make sure /OVERRIDE=ACCESS was specified and the user
995 1412 2 has privilege. For SSS_NOFILACC, SSS_NOVOLACC return the code
996 1413 2 to the user. In this case the user has no access to the tape volume.
997 1414 2 For a 0 give the user all access. For SSS_NORMAL check the VMS
998 1415 2 protection (whatever that means for files? maybe something in the
999 1416 2 future).
1000 1417 2
1001 1418 2 IF .ACCESS EQ SSS_NOVOLACC
1002 1419 2 OR .ACCESS EQ SSS_NOFILACC
1003 1420 2 THEN ERR_EXIT(.ACCESS);
1004 1421 2
1005 1422 2 IF .ACCESS EQ SSS_FILACCERR
1006 1423 2 THEN
1007 1424 3 BEGIN
1008 1425 3 IF NOT .INIT OPTIONS[OPT_OVR_ACC]
1009 1426 3 THEN ERR_EXIT(.ACCESS);
1010 1427 3 IF NOT .PRIVILEGE_MASK[PRVSV_VOLPRO]
1011 1428 3 THEN ERR_EXIT(.ACCESS);
1012 1429 3 ACCESS = SSS_NORMAL;
1013 1430 2 END;
1014 1431 2
1015 1432 2
1016 1433 2 RETURN 0; ! valid to rewrite the ANSI TAPE
1017 1434 1 END; ! end of routine READ_VOLLABLES

```

.EXTRN SYSSBINTIM, SYSSGETTIM

OFFC 00000 READ_VOLLABELS:

				.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11
5B	00000000G	00	9E 00002	MOVAB	SYSSMTAACCESS, R11
5A	0000G	CF	9E 00009	MOVAB	GET RECORD, R10
59	0000G	CF	9E 0000E	MOVAB	PROCESS UIIC, R9
58	00000000G	00	9E 00013	MOVAB	SYSSQIO#, R8
57	00000000G	9F	9E 0001A	MOVAB	2#SYSSCMKRNL, R7
56	00000000G	00	9E 00021	MOVAB	LIB\$STOP, R6
55	0000'	CF	9E 00028	MOVAB	ACCESS, R5
5E		28	C2 0002D	SUBL2	#40, SP
			7E 7C 00030	CLRQ	-(SP)
			7E 7C 00032	CLRQ	-(SP)
7E	50	8F	9A 00034	MOVZBL	#80, -(SP)
		9C	A5 9F 00038	PUSHAB	ANSI LABEL
			7E 7C 0003B	CLRQ	-(SP)
		EC	A5 9F 0003D	PUSHAB	IO STATUS
	0000G	21	DD 00040	PUSHL	#33
		CF	DD 00042	PUSHL	CHANNEL
		7E	D4 00046	CLRL	-(SP)

1174

1229

68	OC	FB	00048		CALLS	#12, SYSSQIOW					
54	50	D0	0004B		MOVL	RO, STATUS					
04	54	E9	0004E		BLBC	STATUS, 1\$					
54	EC	A5	3C	00051	MOVZWL	IO_STATUS, STATUS					1230
FC	AS	69	D0	00055	18:	MOVL	PROCESS_UIC, VOLUME_UIC				
		F8	A5	D4	00059	CLRL	VOLUME_PROT				1236
00000838	40	54	E8	0005C		BLBS	STATUS, 4\$				1237
	8F	37	D1	0005F		CMPL	STATUS, #2104				1243
000002D4	8F	54	D1	00068		BEQL	4\$				1250
		36	12	0006F		CMPL	STATUS, #724				
		7E	7C	00071		BNEQ	5\$				1258
		7E	7C	00073		CLRQ	-(SP)				
		7E	7C	00075		CLRQ	-(SP)				
		7E	7C	00077		CLRQ	-(SP)				
		EC	A5	9F	00079	PUSHAB	IO_STATUS				
			24	DD	0007C	PUSHL	#38				
0000G		CF	DD	0007E		PUSHL	CHANNEL				
		7E	D4	00082		CLRL	-(SP)				
68	OC	FB	00084		CALLS	#12, SYSSQIOW					
54	50	D0	00087		MOVL	RO, STATUS					
07	54	E9	0008A		BLBC	STATUS, 2\$					1259
54	EC	A5	3C	0008D		MOVZWL	IO_STATUS, STATUS				
05	54	E8	00091		BLBS	STATUS, 3\$					1260
		54	DD	00094	28:	PUSHL	STATUS				
FE84	66	01	FB	00096		CALLS	#1, LIBSTOP				1262
	CF	00	FB	00099	38:	CALLS	#0, DEFAULT_CHAR				1265
314C4F56	8F	9C	A5	D1	0009F	RET					1270
		01	13	000A7	48:	CMPL	ANSI_LABEL, #827084630				
		04	04	000A9	58:	BEQL	6\$				
						RET					
08	A5	EB	A5	9A	000AA	68:	MOVZBL	ANSI_LABEL+79, LABEL_VER			1274
08	A5		30	C2	000AF	SUBL2	#48, LABEL_VER				
		OC	A5	DD	000B3	PUSHL	UCB				1283
			01	DD	000B6	PUSHL	#1				
		4400	8F	BB	000B8	PUSHR	#^M<R10, SP>				
04	67		04	FB	000BC	CALLS	#4, SYS\$CMKRNL				
	A5		50	D0	000BF	MOVL	RO, CURRENT_RECORD				
			7E	7C	000C3	CLRQ	-(SP)				1290
			7E	D4	000C5	CLRL	-(SP)				
		08	A5	DD	000C7	PUSHL	LABEL VER				
			69	DD	000CA	PUSHL	PROCESS_UIC				
		9C	A5	9F	000CC	PUSHAB	ANSI_LABEL				
6B	06	FB	000CF		CALLS	#6, SYSSMTACCESS					
65	50	D0	000D2		MOVL	RO, ACCESS					
	OC	A5	DD	000D5		PUSHL	UCB				1292
		01	DD	000D8		PUSHL	#1				
		4400	8F	BB	000DA	PUSHR	#^M<R10, SP>				
67	04	FB	000DE		CALLS	#4, SYS\$CMKRNL					
54	50	D0	000E1		MOVL	RO, STATUS					
54	04	A5	D1	000E4		CMPL	CURRENT_RECORD, STATUS				1293
		08	13	000E8		BEQL	7\$				
7E	0224	8F	3C	000EA		MOVZWL	#548, -(SP)				1294
66	01	FB	000EF		CALLS	#1, LIBSTOP					
50	65	D0	000F2		MOVL	ACCESS, RO					
8F	50	D1	000F5		CMPL	RO, #8868					1304
	09	13	000FC		BEQL	8\$					

000022AC	8F	50 D1 000FE	CMPL R0, #8876	: 1305
		05 12 00105	BNEQ 9S	: 1306
		50 DD 00107	PUSHL R0	: 1308
0000009C	66	01 FB 00109	CALLS #1, LIB\$STOP	: 1311
	8F	65 D1 0010C	CMPL ACCESS, #156	: 1312
05	0000G	18 12 00113	BNEQ 12S	: 1313
	CF	06 E0 00115	BBS #6, INIT_OPTIONS+3, 10S	: 1314
		65 DD 0011B	PUSHL ACCESS	: 1315
05	F4	01 FB 0011D	CALLS #1 LIB\$STOP	: 1316
	B5	15 E0 00120	BBS #21 APRIVILEGE_MASK, 11S	: 1317
		65 DD 00125	PUSHL ACCESS	: 1318
	66	01 FB 00127	CALLS #1, LIB\$STOP	: 1319
	65	01 DD 0012A	MOVL #1, ACCESS	: 1320
		9C A5 9F 0012D	PUSHAB ANSI LABEL	: 1321
		69 DD 00130	PUSHL PROCESS_UIC	: 1322
		F8 A5 9F 00132	PUSHAB VOLUME_PROT	: 1323
		FC A5 9F 00135	PUSHAB VOLUME_UIC	: 1324
0000G	CF	04 FB 00138	CALLS #4, TAPE OWN_PROT	: 1325
	54	50 DD 0013D	MOVL R0, STATUS	: 1326
	12	65 E9 00140	BLBC ACCESS, 13S	: 1327
	12	54 E8 00143	BLBS STATUS, 14S	: 1328
	OD	CF E8 00146	BLBS INIT_OPTIONS+5, 14S	: 1329
	7E	226C 8F 3C 00148	MOVZWL #8812, -(SP)	: 1330
	66	01 FB 00150	CALLS #1, LIB\$STOP	: 1331
		03 11 00153	BRB 14S	: 1332
		F8 A5 D4 00155	CLRL VOLUME PROT	: 1333
B4	A5	0A 29 00158	CMPC3 #10, STARID, ANSI_LABEL+24	: 1334
		05 12 0015F	BNEQ 15S	: 1335
	52	01 90 00161	MOVB #1 VMS_TAPE	: 1336
		02 11 00164	BRB 16S	: 1337
		52 94 00166	CLR8 VMS_TAPE	: 1338
		7E 7C 00168	CLRQ -(SP)	: 1339
		7E 7C 0016A	CLRQ -(SP)	: 1340
	7E	50 8F 9A 0016C	MOVZBL #80 -(SP)	: 1341
		9C A5 9F 00170	PUSHAB ANSI LABEL	: 1342
		7E 7C 00173	CLRQ -(SP)	: 1343
		EC A5 9F 00175	PUSHAB IO_STATUS	: 1344
		21 DD 00178	PUSHL #33	: 1345
		0000G CF DD 0017A	PUSHL CHANNEL	: 1346
		7E D4 0017E	CLRL -(SP)	: 1347
		OC FB 00180	CALLS #12, SYSSQIOW	: 1348
		50 DD 00183	MOVL R0, STATUS	: 1349
	54	54 E9 00186	BLBC STATUS, 17S	: 1350
	07	54 3C 00189	MOVZWL IO_STATUS STATUS	: 1351
	54	54 E8 0018D	BLBS STATUS, 18S	: 1352
00000838	OA	54 D1 00190	CMPL STATUS, #2104	: 1353
	8F	01 13 00197	BEQL 18S	: 1354
		04 00199	RET	: 1355
		52 E9 0019A	BLBC VMS_TAPE, 19S	: 1356
324C4F56	20	9C A5 D1 0019D	CMPL ANSI_LABEL, #843861846	: 1357
	8F	16 12 001A5	BNEQ 19S	: 1358
		9C A5 9F 001A7	PUSHAB ANSI_LABEL	: 1359
		69 DD 001AA	PUSHL PROCESS_UIC	: 1360
		F8 A5 9F 001AC	PUSHAB VOLUME_PROT	: 1361
		FC A5 9F 001AF	PUSHAB VOLUME_UIC	: 1362
0000G	CF	04 FB 001B2	CALLS #4, PROCESS_VOL2_LABEL	: 1363
	03	65 E8 001B7	BLBS ACCESS, 19S	: 1364

		F8	A5	D4	001BA		CLRL	VOLUME PROT		1369	
		9C	A5	D1	001BD	19\$:	CMPL	ANSI_LABEL, #827475016			
			A1	12	001C5		BNEQ	16\$			
		C8	A5	9F	001C7		PUSHAB	ANSI LABEL+47		1375	
		10	AE	9F	001CA		PUSHAB	REGDATE			
	0000G	CF		02	FB	001CD	CALLS	#2, CONVDATE_J2R			
		2F		50	E9	001D2	BLBC	R0, 20\$			
		18	AE	0C	AE	001D5	MOVL	#12, DESCRIPTOR		1378	
		1C	AE	20	90	001D9	MOVAB	REGDATE, DESCRIPTOR+4		1379	
		17	AE	20	AE	001DE	MOVB	#32, REGDATE+11		1380	
				1C	9F	001E3	PUSHAB	DATE		1381	
	00000000G	00		AE	02	FB	001E5	PUSHAB	DESCR		
				02	FB	001E8	CALLS	#2, SYSSBINTIM			
	00000000G	00		5E	DD	001EF	PUSHL	SP		1382	
				01	FB	001F1	CALLS	#1, SYSSGETTIM			
				5E	DD	001F8	PUSHL	SP		1383	
	0000G	CF		24	AE	001FA	PUSHAB	DATE			
				02	FB	001FD	CALLS	#2, CALDAYNO			
				05	11	00202	BRB	21\$		1375	
				6E	D4	00204	20\$:	CLRL	TODAY		
				20	AE	00206		CLRL	DATE		
				0D	18	0020D	21\$:	CMPL	DATE, TODAY		
	07	0000G	CF	03	E0	0020F		BLEQU	22\$		
		7E	B4	8F	9A	00215	BBS	#3, INIT OPTIONS+3, 22\$			
		66	01	FB	00219		MOVZBL	#180, -(SP)		1388	
			OC	A5	DD	0021C	CALLS	#1, LIBSTOP			
				01	DD	0021F	PUSHL	UCB		1397	
				01	DD	00221	PUSHL	#1			
				8F	BB	00221	PUSHR	#^M<R10 SP>			
				04	FB	00225	CALLS	#4, SYSSCMKRNL			
				50	DO	00228	MOVL	R0, CURRENT_RECORD			
				01	DD	0022C	PUSHL	#1		1404	
				7E	7C	0022E	CLRL	-(SP)			
				08	A5	DD	PUSHL	LABEL VER			
				69	DD	00233	PUSHL	PROCESS UIC			
				9C	A5	9F	PUSHAB	ANSI LABEL			
				06	FB	00238	CALLS	#6, SYSSMTACCESS			
				50	DO	0023B	MOVL	R0, ACCESS			
				65	A5	DD	PUSHL	UCB		1406	
				0C	A5	DD	0023E	PUSHL	#1		
				01	DD	00241	PUSHR	#^M<R10 SP>			
				4400	8F	BB	CALLS	#4, SYSSCMKRNL			
				67	04	FB	MOVL	R0, STATUS			
				54	50	DO	CMPL	CURRENT_RECORD, STATUS			
				54	04	A5	BEQL	23\$		1407	
				08	13	00251	MOVZWL	#548, -(SP)			
				7E	0224	8F	CALLS	#1, LIBSTOP			
				66	01	FB	ACCES	R0		1408	
				50	65	DO	R0	#8868			
	000022A4	8F		50	D1	0025E	CMPL	24\$		1418	
				09	13	00265	BEQL	R0			
	000022AC	8F		50	D1	00267	CMPL	#8876		1419	
				05	12	0026E	BNEQ	25\$			
				50	DO	00270	PUSHL	R0		1420	
				66	01	FB	CALLS	#1, LIBSTOP			
	0000009C	8F		65	D1	00272	CMPL	ACCESS, #156		1422	
				18	12	0027C	BNEQ	28\$			
				06	E0	0027E	BBS	#6, INIT_OPTIONS+3, 26\$		1425	

I 16

16-Sep-1984 01:50:56

14-Sep-1984 12:35:18

VAX-11 Bliss-32 V4.0-742

DISK\$VMSMASTER:[INIT.SRC]INITAP.B32;1

Page 37
(5)

		65	DD 00284	PUSHL	ACCESS	: 1426
05	F4	66	01 FB 00286	CALLS	#1 LIB\$STOP	: 1427
		85	15 EO 00289	26\$: BBS	#21 APRIVILEGE_MASK, 27\$: 1428
			65 DD 0028E	PUSHL	ACCESS	: 1429
		66	01 FB 00290	CALLS	#1, LIB\$STOP	: 1434
		65	01 DO 00293	27\$: MOVL	#1, ACCESS	
			04 00296	28\$: RET		

; Routine Size: 663 bytes, Routine Base: \$CODE\$ + 04E7

```
1019      1435 1 ROUTINE CHECK_PROT(VOL_PROT,VOL_UIC) =
1020      1436 1
1021      1437 1 | ++
1022      1438 1
1023      1439 1 | FUNCTIONAL DESCRIPTION:
1024      1440 1 |   this routine check volume protection
1025      1441 1
1026      1442 1 | CALLING SEQUENCE:
1027      1443 1 |   CHECK_PROT(ARG1,ARG2)
1028      1444 1
1029      1445 1 | INPUT PARAMETERS:
1030      1446 1 |   ARG1 - volume protection
1031      1447 1 |   ARG2 - volume owner UIC
1032      1448 1
1033      1449 1 | IMPLICIT INPUTS:
1034      1450 1 |   PROCESS_UIC      - UIC of the current process
1035      1451 1 |   PRIVILEGE_MASK  - mask of privileges that the user has
1036      1452 1 |   INIT_OPTIONS     - init options bitvector
1037      1453 1
1038      1454 1 | OUTPUT PARAMETERS:
1039      1455 1 |   none
1040      1456 1
1041      1457 1 | IMPLICIT OUTPUTS:
1042      1458 1 |   none
1043      1459 1
1044      1460 1 | ROUTINE VALUE:
1045      1461 1 |   SSSNORMAL - if users has the needed priviledges
1046      1462 1 |   SSSNOPRIV - if users does not have the needed priviledges
1047      1463 1
1048      1464 1 | SIDE EFFECTS:
1049      1465 1 |   none
1050      1466 1
1051      1467 1 | USER ERRORS:
1052      1468 1 |   none
1053      1469 1
1054      1470 1 | --
1055      1471 1
1056      1472 2 BEGIN
1057      1473 2
1058      1474 2 MAP
1059      1475 2 |   PROCESS_UIC      : VECTOR [ 2, WORD ], ! the process UIC
1060      1476 2 |   VOL_PROT        : BITVECTOR,
1061      1477 2 |   VOL_UIC         : VECTOR [ 2, WORD ];
1062      1478 2
1063      1479 2 EXTERNAL
1064      1480 2 |   EXE$GL_SYSUIC  : REF BBLOCK ADDRESSING_MODE ( ABSOLUTE );
1065      1481 2
1066      1482 2 LITERAL
1067      1483 2 |   NOT_GROUP_WRITE = 9; ! the group write disable bit
1068      1484 2 |   NOT_WORLD_WRITE = 13; ! the world write disable bit
1069      1485 2
1070      1486 2
1071      1487 2 |   check if the user has write access to the tape
1072      1488 2
1073      1489 2 IF ( .PRIVILEGE_MASK [ PRVSV_BYPASS ] ) OR      ! user has bypass privilege
1074      1490 2 |   ( .PRIVILEGE_MASK [ PRVSV_SYSPRV ] ) OR      ! user has sysprv privilege
1075      1491 2
```

.EXTRN EXE\$GL_SYSUIC

0000 00000 CHECK_PROT:											
0000000G	9F	0000G	CF	50	0000'	CF	D0	00002	.WORD	Save nothing	
				2F	60	1D	E0	00007	MOVL	PRIVILEGE_MASK, R0	
				28	60	1C	E0	00008	BBS	#29, (R0), 1\$	
				27	60	15	E0	0000F	BBS	#28, (R0), 1\$	
				22	05	E1	00013	BBC	#21, (R0), 1\$	1491	
					10	00	ED	00018	CMPZV	#5, VOL_PROT+1, 1\$	1493
					0A	15	15	00023	BLEQ	#0, #16, PROCESS_UIC+2, &EXE\$GL_SYSUIC	1495
					AC	B1	00025	CMPW	1\$	1497	
					08	11	12	0002B	BNEQ	PROCESS_UIC+2, VOL_UIC+2	1500
					AC	CF	B1	0002D	CMPW	2\$	
					05	05	13	00033	BEQL	PROCESS_UIC, VOL_UIC	1501
				04	AC	01	E0	00035	BBS	1\$	
					05	01	D0	0003A	1\$: MOVL	#1, VOL_PROT+1, 2\$	1502
					50	04	0003D	RET	#1, R0	1504	
					50	24	D0	0003E	2\$: MOVL	#36, R0	1508
						04	00041	RET		1510	

; Routine Size: 66 bytes, Routine Base: SCODES + 077E

: 1095 1511 1

```
1097    1512 1 ROUTINE FORMAT_VOL1_VOL2 =
1098    1513 1
1099    1514 1 ++
1100    1515 1
1101    1516 1 FUNCTIONAL DESCRIPTION:
1102    1517 1 This routine formats the volume label one and two, if the user has
1103    1518 1 specified a protection, of an ANSI labeled tape.
1104    1519 1
1105    1520 1 CALLING SEQUENCE:
1106    1521 1 FORMAT_VOL1_VOL2 ()
1107    1522 1
1108    1523 1 INPUT PARAMETERS:
1109    1524 1 none
1110    1525 1
1111    1526 1 IMPLICIT INPUTS:
1112    1527 1 none
1113    1528 1
1114    1529 1 OUTPUT PARAMETERS:
1115    1530 1 none
1116    1531 1
1117    1532 1 IMPLICIT OUTPUTS:
1118    1533 1 none
1119    1534 1
1120    1535 1 ROUTINE VALUE:
1121    1536 1 Value of VOLUME_PROT
1122    1537 1
1123    1538 1 SIDE EFFECTS:
1124    1539 1 The correct infomation gets stuffed into the VOL1 skeleton
1125    1540 1
1126    1541 1 USER ERRORS:
1127    1542 1 none
1128    1543 1
1129    1544 1 --
1130    1545 1
1131    1546 2 BEGIN
1132    1547 2
1133    1548 2 LOCAL
1134    1549 2
1135    1550 2 SPEC,
1136    1551 2 STATUS,
1137    1552 2 VOLUME_PROT,   ! protection for tape
1138    1553 2 VOLUME_UIC;   ! owner of tape
1139    1554 2 BIND
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1560 2
1146    1561 2
1147    1562 2
1148    1563 2
1149    1564 2
1150    1565 2
1151    1566 2
1152    1567 2
1153    1568 2
1134    1549 2
1135    1550 2
1136    1551 2
1137    1552 2
1138    1553 2
1139    1554 2
1140    1555 2
1141    1556 2
1142    1557 2
1143    1558 2
1144    1559 2
1145    1
```

```

1154 1569 2      %ASCII 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa';
1155 1570 2      %ASCII 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa';
1156 1571 2      %ASCII 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa';
1157 1572 2      %ASCII 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa');
1158 1573 2
1159 1574 2
1160 1575 2      ! place the label in the new volume
1161 1576 2
1162 1577 2
1163 1578 2      ! check length of label for volume
1164 1579 2
1165 1580 2      IF .LABEL_STRING [DSCSW_LENGTH] GTRU VL1SS_VOLLBL
1166 1581 2      THEN
1167 1582 2          ERR_EXIT(SSS_MTLBLLONG);
1168 1583 2
1169 1584 2      ! translate the label into upper case and put in 'a' for any non-ANSI
1170 1585 2      a characters found, padded with space in case label from command is
1171 1586 2      less than six characters long
1172 1587 2
1173 1588 2      CHSTRANSLATE ( TRANSLATION_TABLE,
1174 1589 2          .LABEL_STRING [DSCSW_LENGTH],
1175 1590 2          ;LABEL_STRING [DSCSA_POINTER],
1176 1591 2
1177 1592 2          VL1$S_VOLLBL,
1178 1593 2          VOL1[VL1$T_VOLLBL] );
1179 1594 2
1180 1595 2      ! check for non-ANSI 'a' characters
1181 1596 2
1182 1597 2      IF NOT CHSFAIL( CHSFIND_CH ( VL1SS_VOLLBL, VOL1[VL1$T_VOLLBL], 'a'))
1183 1598 2          THEN ERR_EXIT (-INITS_BADVOLLBL );
1184 1599 2
1185 1600 2      ! If the interchange switch is set do not put any VMS specific information on
1186 1601 2      ! to the tape.
1187 1602 2
1188 1603 2      IF NOT .INIT_OPTIONS[OPT_INTERCHG]
1189 1604 2          THEN
1190 1605 2          BEGIN
1191 1606 2
1192 1607 2          ! determine owner and protection of new volume
1193 1608 2
1194 1609 2          IF .INIT_OPTIONS[OPT_PROTECTION]
1195 1610 2              THEN VOL0ME_PROT = .PROTECTION
1196 1611 2              ELSE VOL0ME_PROT = 0;
1197 1612 2
1198 1613 2          IF .INIT_OPTIONS[OPT_OWNER_UIC]
1199 1614 2              THEN VOL0ME_UIC = .OWNER_UIC
1200 1615 2              ELSE VOL0ME_UIC = .PROCESS_UIC;
1201 1616 2
1202 1617 2
1203 1618 2
1204 1619 2
1205 1620 2      FORMAT VOLOWNER(VOL2,.VOLUME_UIC,.VOLUME_PROT);
1206 1621 2
1207 1622 2      THEN
1208 1623 2      BEGIN
1209 1624 2          CH$MOVE(10,STARID,VOL1[VL1$T_SYSCODE]);
1210 1625 2          VOL1[VL1$B_LBLSTDVER] = '4';

```

! end of routine FORMAT_VOL1_VOL2

INITAP
V04-000

C 1
16-Sep-1984 01:50:56 VAX-11 Bliss-32 v4.0-742 Page 43
14-Sep-1984 12:35:18 DISK\$VMSMASTER:[INIT.SRC]INITAP.B32;1 (7)

.ASCII \aa\

.ASCII \aa\

.ASCII \aa\

TRANSLATION_TABLE = P.AAC

.PSECT SCODES,NOWRT,2

03FC 00000 FORMAT_VOL1_VOL2:

			02	11	000A3	BRB	11\$		
			52	D4	000A5	10\$:	CLRL	SPEC	1636
			A7	DD	000A7	11\$:	PUSHL	UCB	1645
			01	DD	000AA		PUSHL	#1	
			5E	DD	000AC		PUSHL	SP	
			CF	9F	000AE		PUSHAB	GET_RECORD	
			04	FB	000B2		CALLS	#4, SY\$CMKRNL	
			50	DO	000B5		MOVL	RO, CURRENT_RECORD	
			02	DD	000B9		PUSHL	#2	
			52	DD	000BB		PUSHL	SPEC	1652
			CF	9A	000BD		MOVZBL	VOL_ACC, -(SP)	
			F0	A7	DD	000C2	FUSHL	LABEL_VER	
			0000G	CF	DD	000C5	PUSHL	PROCESS_UIC	
			7E	D4	000C9		CLRL	-(SP)	
			00000000G	00	06	FB 000CB	CALLS	#6, SY\$MTACCESS	
			F8	A7	50	DO 000D2	MOVL	RO, CHAR	
					F4	A7 DD 000D6	PUSHL	UCB	1654
					01	DD 000D9	PUSHL	#1	
					5E	DD 000DB	PUSHL	SP	
					0000G	CF 9F 000DD	PUSHAB	GET_RECORD	
					04	FB 000E1	CALLS	#4, SY\$CMKRNL	
					50	EC A7 D1 000E4	CMPL	CURRENT_RECORD, STATUS	1655
					50	EC A7 D1 000E4	BEQL	12\$	
					08	13 000E8	MOVZWL	#548, -(SP)	1656
					7E	0224 8F 3C 000EA	CALLS	#1, LIB\$STOP	
					68	01 FB 000EF	MOVB	CHAR, VOL1+10	1657
					06	A7 F8 A7 90 000F2	MOVL	VOLUME_PROT, RO	1662
					50	56 DO 000F7	RET		1664
					04	000FA			

: Routine Size: 251 bytes. Routine Base: \$CODE\$ + 07C0

: 1250 1665 1
: 1251 1666 1 END
: 1252 1667 0 ELUDOM

.EXTRN LIB\$SIGNAL, LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
\$PLITS	276 NOVEC,NOWRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)	
\$OWNB	440 NOVEC, WRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)	
\$CODE\$	2235 NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)	

Library Statistics

File	----- Symbols -----			Pages Mapped	Processing Time
	Total	Loaded	Percent		

INITAP
VO4-000

: _\$255\$DUA28:[SYSLIB]LIB.L32;1

18619

70

E

1

16-Sep-1984 01:50:56
14-Sep-1984 12:35:18

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[INIT.SRC]INITAP.B32;1

Page 45
(7)

00:01.9

COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS:INITAP/OBJ=OBJ\$:INITAP MSRC\$:INITAP/UPDATE=(ENH\$:INITAP)

: Size: 2235 code + 716 data bytes

: Run Time: 00:48.8

: Elapsed Time: 01:37.4

: Lines/CPU Min: 2049

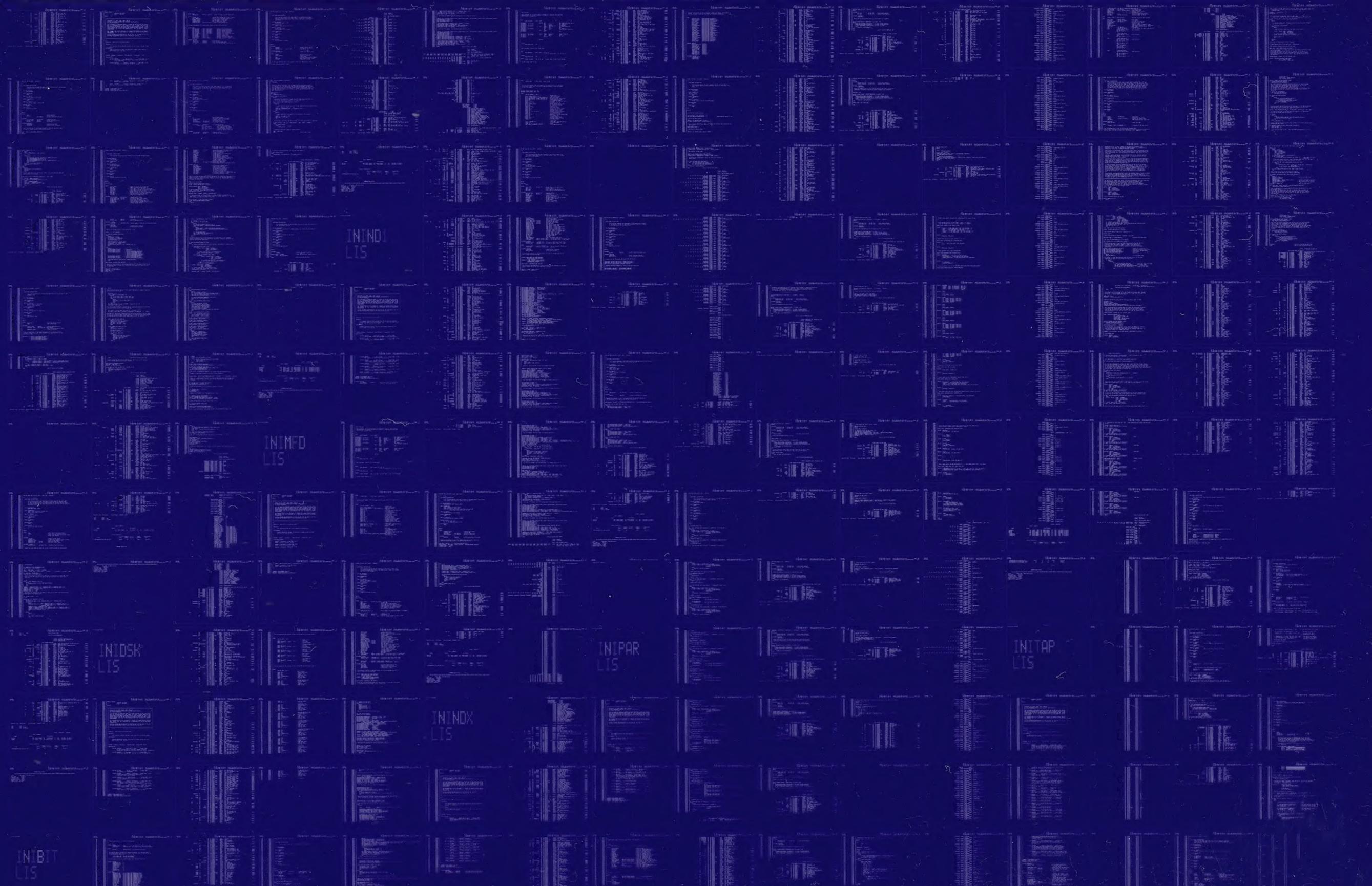
: Lexemes/CPU-Min: 29610

: Memory Used: 365 pages

: Compilation Complete

0187 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



0188 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

INPSMB
MAP

IN\$DEF
SOL

INPSMBMSG
LIS

RSXLBLDF
SOL

INSCREATE
LIS

INSTAL

INSTALL
MAP

IN\$CMO
CLO

INSPREFIX
REQ

INPSMBOLD
CLO

INPSMB
LIS

IN\$OLCMO
CLO

IN\$CMO
LIS

INITIO
LIS

ROHOME
LIS

INPSMB
LIS